DETECTING AND TRACKING MOVING OBJECTS FROM A MOVING PLATFORM

A Thesis
Presented to
The Academic Faculty

by

Chung-Ching Lin

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August 2012

DETECTING AND TRACKING MOVING OBJECTS FROM A MOVING PLATFORM

Approved by:

Professor Marilyn Wolf, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Sudhakar Yalamanchili
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Anthony Yezzi
School of Electrical and Computer Engineering
Georgia Institute of Technology

Professor Hyesoon Kim
School of Computer Science
Georgia Institute of Technology

Professor Justin Romberg
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: 16 April 2012

Dedicated to

My Beloved Family, Shin-Yi Megan Tseng and Erin Heather Lin

ACKNOWLEDGEMENTS

I would like to express my most sincere gratitude to my research advisor, Professor Marilyn Wolf, for her guidance and support through my PhD study. She provided an exciting working environment with many opportunities to develop ideas. It is my best fortune to have been working with her.

I would also like to thank Professor Anthony Yezzi and Professor Justin Romberg for accepting to be in my reading committee. I am also grateful to the other committee members, Professor Sudhakar Yalamanchili and Professor Hyesoon Kim, for accepting this task with enthusiasm.

I would like to express my gratitude to my colleagues, Dongwon Lee, Sehum Kim, Honggab Kim, and Muhammad Umer Tariq. They turned these years at GT into a pleasant time.

I am indebted to my parents, my sisters, and Megan's family. Their affectionate support has always been very important to me. Last, but not least, my deepest love goes to my wife Megan for her loving care, long-standing support and for her great understanding of the working structure of science that has helped walk me through difficult times. I also wish to express my wholehearted thank to my daughter Erin for bringing so much happiness and pleasure into my PhD study life. I could not have completed this work without their love and company. They allowed me to spend most of the time on this thesis. I truly dedicate this dissertation to Megan and Erin.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

Detecting and tracking moving objects are important topics in computer vision research. Classical detecting and tracking methods for steady cameras are not suitable for use with moving cameras because the assumptions for these two applications are different. This thesis aims to develop algorithms that can detect and track moving objects with a non-fixed position camera. To achieve this aim, we analyze the image sequences captured by a moving camera undergoing general 3D rotation and translation. New computer vision algorithms are developed to obtain feasible solutions to the problem without prior camera calibration and classifier training.

The initial step of this research is to develop a new method for estimating camera motion parameters. Based on the estimated camera motion parameters, two methods are developed for detecting moving objects: one based on the Bayesian decision and another based on the belief propagation. The Bayesian decision method uses camera motion parameters to compensate for the camera motion. The background classification rule for every pixel is developed to generate a foreground mask, and then the moving objects can be detected. Another detection method addresses the detection problem by creating a graphical model, which uses the belief propagation algorithm. After camera motion parameters are estimated, feature points in every frame are grouped using a hierarchical clustering algorithm. Then, the related groups between adjacent frames are linked, which results in a graphical model. A belief propagation algorithm is used to transmit the information on this graphical model to find which group is on the moving object.

For moving-object tracking, a classical particle filter works well for tracking moving objects by steady cameras, but it is inadequate for tracking moving objects by moving cameras. Moving cameras often have sudden and shaking movement, which can cause image blur and large position changes, resulting in tracking failures. A new algorithm is proposed to overcome the tracking failure caused by sudden movement, and to track the moving object with correct position and size. A feature-guided particle filter is derived to track moving objects. The Markov Chain Monte Carlo (MCMC) technique is used to implement the filter more efficiently.

In addition to the above detection methods, we develop two on-line detection methods that do not use camera motion parameters for detecting road region and on-road objects. One method uses the feature points and the other uses superpixels. Most existing related approaches operate with a pre-defined class and require a model to be trained in advance. In both our detection approaches, prior trained classifiers are not required. In the approach using feature points, we propose an on-line learning method for detecting the road region and objects on the road. The algorithm learns the key feature points of the road region based on the detected feature points. All the feature points in a frame are classified by the key feature points. The road region is labeled using the classified feature points. Finally, the feature points on the labeled road region are used to detect the objects on the road. In the approach using superpixels, the first step is to apply bottom-up segmentation on the input images to obtain the superpixels. The scene is parsed into less segmented regions by merging similar superpixels. Then, the parsing results are utilized to estimate the road region and detect vehicles on the road using the properties of superpixels. Finally, a tracking method based on superpixel properties is further developed to complete the system.

This research is related to image processing and computer vision. By combining machine learning and pattern recognition techniques, algorithms are developed for a moving camera to detect and track moving objects. To apply these methods with

different types of cameras, only image information is used. Advantages of these methods include avoiding the need for camera calibration and prior training, as well as, the need for information from other sensors, e.g., radar, odometer, GPS, laser, etc.

Experimental results show that these methods demonstrate significant object detecting and tracking performance without further restrictions, and perform effectively in complex environments.

CHAPTER I

INTRODUCTION

Computer vision research includes several important topics, one of which is detecting and tracking moving objects. After many years of research on steady cameras, many sophisticated detecting and tracking algorithms have been developed. Currently, as cameras are becoming less and less expensive, almost everyone has a camera particularly on moving platforms. However, these algorithms for steady cameras are not suitable for moving cameras because the assumptions for both applications differ. Thus, this thesis aims to develop detecting and tracking algorithms for a monocular moving camera without prior camera calibration and classifier training. To achieve this aim, we analyze the image sequences captured by a moving camera undergoing general 3D rotational and translational movement. Typical examples are car-mounted or handheld cameras. In the image sequences, multiple objects move in a static background. In this study, an uncalibrated monocular camera is used. For more general applications, we assume that any intrinsic and extrinsic parameters of the camera are unknown. Besides, the objects' appearances, dimensions, and identities are also unknown. The image sequences we used were shot using a camera held by a human in a moving vehicle. The movement of the camera is unstable because of the uneven road and a shaking human hand.

This thesis work can be applied to several areas, including video surveillance, driving safety assistance, robot navigation, and content-based video coding. Moving-object detection and tracking are the basic steps for further analysis of image sequences from video surveillance. The ability to detect and track moving obstacles can improve the safety of driving and robot navigation. In addition, by segmenting

the motion region, different bit-rates can be applied to moving regions and static backgrounds to achieve more efficient data storage and transmission.

This thesis proposes a new method to estimate camera motion parameters. Based on estimated parameters, machine-learning techniques are used to solve the moving-object detection problem. Two detection algorithms are developed based on estimated camera motion parameters: one based on the Bayesian decision and another based on the belief propagation. Then, this work proposes a tracking algorithm for the application of moving cameras. Further, two additional detection algorithms are developed without estimating camera motion parameters: one based on feature-point analysis and the other on superpixel analysis.

The Bayesian decision method classifies a background and foreground for every pixel to build a foreground mask. In this method, we first detect feature motion vectors and estimate camera motion parameters. Using the camera motion parameters, we compute current frame estimation and then use the detected feature motion vectors to reduce the noise, and generate a foreground template, which is used to update the probability model derived by the Bayesian decision rule. Then, we classify every pixel as a foreground or a background pixel and generate a foreground mask, which is used to detect a moving object. The experiment results of the proposed method are compared with those results of the existing method using multi-view geometric constraints. The experiment results show that the proposed method performs better than the existing method.

The second detection algorithm using camera motion parameters is the belief propagation method, based on a feature-level process instead of a pixel-level process, which decreases computational complexity and memory use. The belief propagation method clusters feature points in every frame based on the different deviations of the detected feature motion vectors and calculated feature motion vectors. Then, the related groups of adjacent frames are linked. The problem of moving-object

detection is formulated as a graphical model, and the belief propagation is applied to this model. The belief value of every group is calculated and used to further classify moving objects. The benefit of the belief propagation method is that it has lower computational complexity and less memory use than the Bayesian decision method, but the Bayesian decision method provides more precise sizes and positions of moving objects than the belief propagation method.

For moving-object tracking, a classical particle filter works well for tracking moving objects by steady cameras, but it is inadequate for tracking moving objects by moving cameras. Moving cameras often have sudden movement and shaking, which cause image blur and large position changes, resulting in tracking failures. A new algorithm is proposed to overcome the tracking failure caused by sudden movement and to track the moving objects with correct positions and sizes. A feature-guided particle filter is derived to track moving objects. The Markov Chain Monte Carlo (MCMC) technique is used to implement the filter more efficiently.

In addition to the previously mentioned detection methods, we develop two methods that do not use camera motion parameters for dynamic scene analysis. In the dynamic scene analysis area, researchers recently have successfully built classifiers for detecting objects, but most approaches operate with a pre-defined class and require classifiers to be trained in advance. In our detection methods, pre-trained classifiers are not required. In the method using feature points, an on-line learning method is proposed for detecting the road region and on-road objects by analyzing the image sequences captured by a monocular camera on a moving platform. In this approach, the key feature points of the road region are learned based on the detected and matched feature points between adjacent frames without using camera-intrinsic parameters or camera-motion parameters. Then, all the feature points in a frame are classified as on-road or non-road using the key feature points. The road region is labeled using the classified feature points. Finally, the feature points on the labeled road region are

used to detect the on-road objects.

In the detecting method using superspixels, we present a system with a novel approach to multi-vehicle detection. In this approach, a bottom-up segmentation is first applied on the input images to get the superpixels. The scene is parsed into less segmented regions by merging similar neighboring superpixels. Then, we estimate the road region and detect on-road vehicles using the superpixels' properties of the new parsing result. Finally, tracking is achieved and could be used to provide feedback for further guiding vehicle detection in future frames.

This research is related to image processing and computer vision. By combining machine-learning and pattern-recognition techniques, algorithms are developed for a moving camera to detect and track moving objects. To apply these methods with different types of cameras, only image information is used. The advantages of these methods include avoiding the need for camera calibration and prior training, as well as, the need for information from other sensors, e.g., radar, odometer, GPS, laser, etc.

Experimental results show that these methods demonstrate significant object detecting and tracking performance without further restrictions, and perform effectively in complex environments.

1.1 Contributions

Existing methods for moving camera detecting impose serious constraints, thus the application is limited. In our detection methods using camera motion parameters, a new formula for estimating camera motion parameters is derived. A background probability model is developed for the application of a moving camera based on a Bayesian decision rule. The probability update scheme adjusts the probability models using the information in every incoming frame. The proposed detection algorithm can process noises effectively and demonstrates better detecting performance than

the existing methods in the complex environment. There is no need to impose initial assumptions or to apply future frame information. Thus, the method could be generally applied to the detecting problem. In the belief propagation method, we address the detection problem by creating a graphical model and transmitting the information along the edges of the graphical model. Since only group information is processed, the belief propagation method reduces the computational complexity and memory use.

In moving-object tracking, sudden camera or object motion is the typical problem that causes tracking performance to sharply deteriorate. It is inadequate to use classical recursive Bayesian estimation to track moving objects observed by a rapid-moving and unstable camera since the classical method cannot resolve the sudden motion problem. We develop a robust and unconstrained Markov Chain Monte Carlo based feature-guided particle filtering algorithm to overcome the tracking failure issues. The method achieves accurate tracking performance in object sizes and positions without the assistance of foreground segmentation and performs well in severe tracking environments.

In dynamic scene analysis, most existing methods for moving-camera detection operate with pre-defined classifiers and require a model to be trained in advance. One of the main disadvantages of the off-line learning method is the need to collect and train data in advance for a specific application. When an object size or shape does not appear in the training data set, it would cause a detection failure. To adapt to the variety of environments and object types, we present two new methods for detecting road regions and on-road objects without using a prior trained classifier. These methods on-line analyze the properties of the road region, which allow these method to adapt to constantly changing environments. With this property, the proposed algorithms can solve the existing problems and generally be applied to the real-world system.

This thesis provides feasible solutions to the detection and tracking problems with fewer constraints. In the development of algorithms, we assume that any intrinsic or extrinsic parameters of the camera are unknown and no annotated data is used for classifier training. The only information we use is the input image sequences captured by a monocular moving camera. Therefore, the proposed algorithms can be successfully applied to different types of cameras and a variety of environments.

1.2 Thesis Organization

This thesis is structured as follows. Chapter 2 is devoted to introducing the fundamental techniques for analyzing image sequences captured by a moving camera. Chapter 3 presents two algorithms for moving-object detection without prior camera calibration. In Chapter 4, an algorithm is presented for moving-object tracking using an MCMC-based feature-guided particle filter. Chapter 5 presents two algorithms of dynamic scene analysis for road region and on-road object detection that do not require prior classifier training. Finally, conclusions and proposals for future extensions to this work are provided in Chapter 6.

CHAPTER II

MULTIPLE IMAGE REGISTRATION

This chapter presents the method of multi-image registration, aligning multiple image from different views. The registration process estimates the geometric relationship between the pixels across multiple images from different views. The geometric relationship helps the prediction of the corresponding positions for pixels in different frames. One application of the image registration process is to compensate for the camera motion.

The first step in estimating the geometric relationship is to measure the pixel movement between two frames. Not every pixel's movement is reliable; therefore this chapter presents the methods that obtain the reliable measurement of a pixel's correspondence, that is, feature extraction and matching. After obtaining the reliable measurement of a pixel's movement, the camera model is presented. Then, the geometric relationship is introduced.

2.1 Feature Extraction and Matching

This section discusses methods for detecting and matching feature points. A feature point refers to a pixel that has a well-defined position and can be robustly detected. Feature extracting and matching techniques are widely used in motion detection, image registration, image mosaicking, object recognition, 3D reconstruction, and augment reality.

2.1.1 Harris Corner

The Harris corner detector [27] is a commonly used feature point detector and is strong invariance to rotation and scale. The Harris corner detector performs well on

the edges and corners of man-made objects, e.g., buildings and vehicles. The Harris corner detector is based on the autocorrelation with respect to direction directly, instead of using shifted patches.

Let I be the be the grayscale 2D image used in the derivation. In this image, an image patch over the area (u, v) and shifting it by (x, y) is considered to compute a autocorrelation. The autocorrelation function $E(x, y)$ is calculated with the weighted sum of squared differences (SSD) between two patches and is given by

$$E(x,y) = \sum_{u,v} w(u,v)(I(x+u, y+v) - I(x+u+\Delta x, y+v+\Delta y))^2. \quad (1)$$

$I(x+u+\delta x, y+v+\delta y)$ can be approximated by a Taylor expansion truncated to the first-order terms,

$$I(x+u+\delta x, y+v+\delta y) \approx I(x+u, y+v) + [I_x(x+u, y+v)\ I_y(x+u, y+v)][\Delta x\ \Delta y]^T, \quad (2)$$

where $I_x(x, y)$ and $I_y(x, y)$ are the partial derivatives of I in x and y, respectively.

Substituting approximation Equation (2) in to Equation (1) yields

$$\begin{aligned}
E(x,y) &= \sum_{u,v} w(u,v)(I(x+u, y+v) - I(x+u+\Delta x, y+v+\Delta y))^2 && (3)\\
&\approx \sum_{u,v} w(u,v)(I(x+u, y+v) - I(x+u, y+v) \\
&\quad -[I_x(x+u, y+v)\ I_y(x+u, y+v)][\Delta x\ \Delta y]^T)^2 && (4)\\
&= \sum_{u,v} w(u,v)(-[I_x(x+u, y+v)\ I_y(x+u, y+v)][\Delta x\ \Delta y]^T)^2 && (5)\\
&= \sum_{u,v} w(u,v)([I_x(x+u, y+v)\ I_y(x+u, y+v)][\Delta x\ \Delta y]^T)^2 && (6)\\
&= [\Delta x\ \Delta y] \sum_{u,v} w(u,v)([I_x(x+u, y+v)\ I_y(x+u, y+v)]^T \\
&\quad [I_x(x+u, y+v)\ I_y(x+u, y+v)])[\Delta x\ \Delta y]^T && (7)\\
&= [\Delta x\ \Delta y] M(x,y) [\Delta x\ \Delta y]^T, && (8)
\end{aligned}$$

where

$$M(x,y) = \sum_{u,v} w(u,v)$$

$$= \begin{bmatrix} (I_x(x+u,y+v))^2 & I_x(x+u,y+v)I_y(x+u,y+v) \\ I_x(x+u,y+v)I_y(x+u,y+v) & (I_y(x+u,y+v))^2 \end{bmatrix} \quad (9)$$

$$= \begin{bmatrix} A & C \\ C & B \end{bmatrix}. \quad (10)$$

The matrix M is a Harris matrix.

If the window $w(u,v)$ is binary and rectangular, the response will be noisy. Harris and Stephens suggest using a smooth circular window, e.g., a Gaussian:

$$w(u,v) = \exp -(u^2+v^2)/(2\sigma^2). \quad (11)$$

A corner is characterized by a large variation of E in all directions of the vector $[x\ y]$. Let α and β be the eigenvalues of M. By analyzing the eigenvalues of M, the corner characterization can be expressed in the following way:

- If $\alpha \approx 0$ and $\beta \approx 0$, this pixel (x,y) has no features of interest.
- If $\alpha \approx 0$ and β has some large positive value, then an edge is found.
- If α and β have large positive values, then a corner is found.

The exact computation of the eigenvalues is computationally expensive because it requires the computation of a square root. Harris and Stephens suggest the following corner response function to evaluate corner strength.

$$R = det(M) - k \cdot trace(M)^2, \quad (12)$$

where

$$trace(M) = \alpha + \beta = A + B \quad (13)$$

$$det(M) = \alpha * \beta = AB - C^2, \quad (14)$$

and k is a tunable sensitivity parameter. Therefore, unlike the Kanade-Tomasi corner detector, the algorithm does not have to actually compute the eigenvalue decomposition of the matrix M. The determinant and trace of M are sufficient to find corners.

2.1.2 Scale-Invariant Feature Transform (SIFT)

SIFT [51] was developed by David Lowe and performs invariant feature detection. Extraction of keypoints is performed using four steps: 1) Scale-space extrema detection 2) Keypoint localization 3) Orientation assignment 4) Keypoint descriptor. First, SIFT uses scale-space extrema detection to detect the locations of feature points. The image is convolved with Gaussian filters at different scales, and then the differences of successive Gaussian-blurred images are taken. Keypoints are then taken as maxima/minima of the Difference of Gaussians (DoG) that occur at multiple scales. A DoG image $D(x, y, \sigma)$ is given by

$$D(x, y, \sigma) = L(x, y, k_i\sigma) - L(x, y, k_j\sigma), \tag{15}$$

where $L(x, y, k\sigma)$ is the convolution of the original image $I(x, y)$ with the Gaussian blur $G(x, y, k\sigma)$ at scale $k\sigma$, i.e.,

$$L(x, y, k\sigma) = G(x, y, k\sigma) * I(x, y) \tag{16}$$

Therefore, a DoG image between scales $k_i\sigma$ and $k_j\sigma$ is the difference of the Gaussian-blurred images at scales $k_i\sigma$ and $k_j\sigma$. For scale-space extrema detection in the SIFT algorithm, the image is first convolved with Gaussian blurs at different scales. The convolved images are grouped by octave, and the value of k_i is selected so that a fixed number of convolved images per octave can be obtained. Then the Difference-of-Gaussian images are taken from adjacent Gaussian-blurred images per octave.

When DoG images have been obtained, keypoints are identified as the local minima/maxima of the DoG images across scales. This is done by comparing each pixel

in the DoG image to its eight neighbors at the same scale and nine corresponding neighboring pixels in each of the neighboring scales. If the pixel value is the maximum or minimum among all compared pixels, it is selected as a candidate keypoint.

Next step is keypoint localization. Scale-space extrema detection produces too many keypoint candidates, some of which are unstable. The next step in the algorithm is to perform a detailed fit to the nearby data for accurate location, scale, and ratio of principal curvatures. This information is used to evaluate the contrast and the localization of the points. If the points have low contrast or are poorly localized along an edge, the points are rejected.

For each candidate keypoint, interpolation of nearby data is used to accurately determine its position. The interpolation is done using the quadratic Taylor expansion of the Difference-of-Gaussian scale-space function, $D(x, y, \sigma)$, with the candidate keypoint as the origin. This Taylor expansion is given by:

$$D(\chi) = D + \frac{\partial D^T}{\partial \chi}\chi + \frac{1}{2}\chi^T \frac{\partial^2 D}{\partial \chi^2}\chi, \tag{17}$$

where D and its derivatives are evaluated at the candidate keypoint and $\chi = (x, y, \sigma)$. If the value of the second-order Taylor expansion $D(\chi)$ is too small, the candidate keypoint is considered as a point with low contrast and is discarded.

For poorly defined peaks in the DoG function, the principle curvature across the edge would be much larger than the principal curvature along it. The principal curvatures can be computed from a 2x2 Hessian matrix, H:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}. \tag{18}$$

The eigenvalues of H are proportional to the principal curvatures of D. Let the α and β be the eigenvalues of H. The ratio r of the two eigenvalues is sufficient for SIFT's purposes.

$$r = \alpha/\beta, \tag{19}$$

where $\alpha > \beta$.

$$R = trace(H)^2/det(H) = (r+1)^1/r \tag{20}$$

If the keypoint is poorly localized, R for the candidate keypoint will be larger. A threshold r_{th} is used to reject the poorly localized keypoints.

After localizing keypoints, one or more orientations are assigned to each key point location based on local image gradient directions. With the orientation assigned, keypoint descriptors are extracted from image data that has been transformed relative to the assigned orientation, scale, and location for each keypoint. This process provides rotation and scale invariances. For an image sample $L(x, y)$ at the closest scale, the gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, are:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{21}$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right). \tag{22}$$

A histogram is built from the gradient orientations of the neighbors of a keypoint. The highest peak in the histogram and all other peaks within 80% of the highest peak are set as the orientation of the keypoint.

A keypoint descriptor is created by computing orientation histograms in a region around the keypoint location. A set of orientation histograms is created on 4x4 pixel neighborhoods with 8 bins each. These histograms are computed from magnitude and orientation values of samples in a 16x16 region around the keypoint. Therefore, each histogram contains samples from a 4x4 sub-region of the original neighborhood region. The descriptor is a vector with 128 elements (4x4x8).

2.1.3 Speeded-Up Robust Features (SURF)

SURF [4] was developed by Bay et al. to improve the speed of the interest point detector, the descriptor generation, and the matching. SURF uses a very basic Hessian-matrix approximation for feature point detection. At a point $p = (x, y)$ in an image

I, Hessian matrix $H(p,\sigma)$ in p at scale σ is defined as follows

$$H(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix}, \tag{23}$$

where $L_{xx}(p,\sigma)$ is the convolution of the Gaussian second-order derivative $\frac{\partial^2}{\partial x^2}g(\sigma)$ with the image I in point x, and similarly for $L_{xy}(p,\sigma)$ and $L_{yy}(p,\sigma)$.

Bay et al. approximate the Hessian matrix with box filters. The box filters approximate second order Gaussian derivatives and the filtering can be performed using integral images with a very low computational complexity. And, the calculation time is independent of the filter size.

Let D_{xx}, D_{yy}, and D_{xy} be the approximations of L_{xx}, L_{yy}, and L_{xy}, respectively. The weights w applied to the rectangular regions are kept simple for computational efficiency.

$$det(H) \approx D_{xx}D_{yy} - (wD_{xy})^2. \tag{24}$$

The relative weight w of the filter responses is used to balance the expression for the Hessian's determinant.

The filter responses are further normalized with respect to their size, which guarantees a constant Frobenius norm for any filter size. With the Frobenius norm remaining constant for the box filters at any size, the filter responses are scale normalized and require no further weighting.

Bay et al. use up-scaling the filter size to analyze the scale space instead of implementing scale spaces as an image pyramid. The construction of the scale space starts with the 9x9 filter. Then, filters with sizes 15x15, 21x21, and 27x27 are applied.

To localize the feature points, Bay et al. use a fast variant on a non-maximum suppression in a 3x3x3 neighborhood, and then interpolate the maxima of the determinant of the Hessian matrix in scale and image space.

Similar to the gradient information extracted by SIFT, the SURF descriptor describes the distribution of the intensity content within the interest point neighborhood. Unlike SIFT, the SURF descriptor builds on the distribution of the first-order Haar wavelet responses in the x and y direction rather than the gradient. For first order Haar wavelet responses, integral images are used to reduce the computational complexity, and 64 dimensions is used.

The first step for extracting the descriptor is to construct a square region that is centered on the feature point and is oriented along the orientation selected. The size of this window is 20s. Sub-regions (4x4 square) are created inside that region. For each sub-region, Haar wavelet responses are computed at 5x5 regularly spaced sample points. In SURF, d_x and d_y are defined as the Haar wavelet response in the horizontal and vertical directions, respectively. To increase the robustness toward geometric deformations and localization errors, wavelet response is taken and summed up after weighted with a Gaussian for each sub region. As a result, each sub-region has a four-dimensional descriptor vector v for its underlying intensity structure $v = [\sum d_x \ \sum d_y \ \sum |d_x| \ \sum |d_y|]$. Therefore, the SURF descriptor vector has 64 dimensions (4x4x4). The wavelet responses are invariant to a bias in illumination. The invariant to contrast can be achieved by normalizing the descriptor to one.

Bay et al. show that SURF outperforms SIFT in their experiments and believe that is due to the fact that SURF integrates the gradient information within a subpath, whereas SIFT depends on the orientations of the individual gradients. The integration makes SURF less sensitive to noise.

2.1.4 Line Detection

One of commonly used line detection algorithm is the Hough transform. The Hough transform is a feature-extraction technique used in image analysis and computer vision [19]. The purpose of this technique is to find imperfect instances of objects within a

certain class of shapes by a voting procedure.

A case of Hough transform is the linear transform for detecting straight lines. In the image space, the straight line can be described as $y = mx + b$ and can be graphically plotted for each pair of image points (x, y). In the Hough transform, the main idea is to consider the characteristics of the straight line not as image points (x_1, y_1), (x_2, y_2), etc., but instead, in terms of its parameters, i.e., the slope parameter m and the intercept parameter b. Based on this fact, the straight line $y = mx + b$ can be represented as a point (b, m) in the parameter space. However, one faces the problem that vertical lines give rise to unbounded values of the parameters m and b. For computational reasons, it is therefore better to use a different pair of parameters, denoted r and θ, for the lines in the Hough transform. These are the polar coordinates. The parameter r represents the distance between the line and the origin, while θ is the angle of the vector from the origin to the closest point. Using this parameterization, the equation of the line can be written as

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right) + \left(\frac{r}{\sin\theta}\right), \tag{25}$$

which can be rearranged as $r = x\cos\theta + y\sin\theta$. If $\theta \in [0, 2\pi)$ and $r \geq 0$, the line can be uniquely mapped with a point on the (r, θ) plane. The (r, θ) plane is sometimes called the Hough space for the set of straight lines in two dimensions.

2.2 Camera Model

In [28], Hartley and Zisseman introduced a camera model in matrix forms. A camera model presents a mapping relationship between the 3D world and a 2D image. The central projection of points in space onto a plane is considered. Let the center of projection be the origin of a Euclidean coordinate system, and consider the plane $z = f$, which is called the image plane or focal plane (f is a focal length). Under the pinhole camera model, all the projection lines intersect at the center. A point in space with coordinates $P = (X, Y, Z)^T$ is mapped to the point on the image plane

where a line joining the point P to the center of projection meets the image plane. The point $(X,\ Y,\ Z)^T$ is mapped to the point $(fX/Z,\ fY/Z,\ f)^T$ on the image plane using similar triangles. By ignoring the final image coordinate, the central projection mapping from world to image coordinates can be described as

$$(X, Y, Z)^T \rightarrow (fX/Z, fY/Z)^T. \tag{26}$$

This is a mapping from Euclidean 3D space to Euclidean 2D space.

To express the mapping with a linear form for central projection, the 3D points are represented by homogeneous vectors. Then, the central projection can be written in terms of matrix multiplication as

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{27}$$

Let P and p representing the homogeneous vector $[X\ Y\ Z\ 1]^T$ and the homogeneous vector of a image point, Equation (27), can be written as

$$p_i = \boldsymbol{M_i} P, \tag{28}$$

where M is the camera matrix for the pinhole model of central projection and is defined as

$$M = diag(f, f, 1)[I|0]. \tag{29}$$

Equation (27) assumes that the origin of coordinates in the image plane is at the principal point. In a more general case, the mapping considers the principal point offset

$$(X, Y, Z)^T \rightarrow (fX/Z + o_x, fY/Z + o_y)^T, \tag{30}$$

where $(o_x, o_y)^T$ are the coordinates of the principal point. This mapping can be

expressed in homogeneous coordinates as

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{bmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{31}$$

The concise form of Equation (31) is

$$p = K[I|0]P, \tag{32}$$

where K is the camera calibration matrix

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}. \tag{33}$$

In a more general analysis, the point in space should be expressed in terms of the world coordinate frame. The two coordinate frames are related via a rotation and a translation. Given that $\hat{P}$ is an inhomogeneous 3-element vector representing the coordinates of a point in the world coordinate frame, and $\hat{P}_{cam}$ is the same point in the camera coordinate frame, $\hat{P}_{cam}$ can be written as $\hat{P}_{cam} = R(\hat{P} - \hat{C})$, where $\hat{C}$ represents the coordinates of the camera's center in the world coordinate frame, and R is a 3x3 rotation matrix representing the orientation of the camera coordinate frame. The transformation between the camera and the world coordinate frame can be expressed in a homogeneous form as

$$P_{cam} = \begin{bmatrix} R & -R\hat{C} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} R & -R\hat{C} \\ 0 & 1 \end{bmatrix} P. \tag{34}$$

The mapping equation can be written as

$$p = KR[I| - \hat{C}]P, \tag{35}$$

where P is now in the world coordinate frame. The camera matrix is then expressed as

$$M = K[R|t], \tag{36}$$

where $t = -R\hat{t}$.

Let the number of pixels per unit distance in image corrdinates are m_x and m_y in the x and y directions. The general form of the calibration matrix of a CCD camera is

$$K = \begin{bmatrix} \alpha_x & & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}, \tag{37}$$

where $\alpha_x = fm_x$ and $\alpha_y = fm_y$ represent the focal length of the camera in terms of pixel dimensions in the x and y direction respectively.

By adding more generality, a calibration matrix is considered as the following form:

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix}, \tag{38}$$

where s is referred to as the skew parameter. The skew parameter is zero for most normal cameras.

A camera with the calibration matrix K (Equation (37)) is called a finite projective camera and the camera matrix is expressed as

$$M = KR[I| - \hat{C}]. \tag{39}$$

2.3 Homography

If all the object points lie on a 3D plane, their coordinates in the current image I and the goal image I^2 are related by a "colineation." Assume that a point P lies on a plane π whose normal vector is N. The point expressed in the current camera frame

is related to the goal camera frame by a rotation matrix R and translation vector T as

$$P^{(2)} = RP^{(1)} + T. \tag{40}$$

Let d be the distance of the plane π from the current camera center.

$$d = N^T P^{(1)}. \tag{41}$$

The relation can be expressed as

$$P^{(2)} = (R_2 + \frac{T_2 N^T}{d})P^{(1)}. \tag{42}$$

If the camera intrinsic parameters are known, the image coordinates of the 3D points are given by $p^{(1)} = P^{(1)}/Z^{(1)}$ and $p^{(2)} = P^{(2)}/Z^{(2)}$.

$$p^{(2)} \approx H_{21} p^{(2)}. \tag{43}$$

H_{21} is called the "homography" matrix up to a scale factor.

$$H_{21} = K_2(R_2 + \frac{T_2 N^T}{d})K^{-1}. \tag{44}$$

CHAPTER III

DETECTING MOVING OBJECTS WITH CAMERA MOTION PARAMETERS

Algorithms of moving-object detection are proposed in this chapter. We first introduce an ego-motion estimation method for videos captured by a moving camera on a moving platform. In our approach, an ellipsoid scene shape is applied in the motion model and a complex ego-motion estimation formula is derived. A genetic algorithm is introduced to accurately solve ego-motion parameters. Two detection algorithms, the Bayesian decision method and the belief propagation method, are then developed based on the ego-motion estimation method. The Bayesian decision method is a pixel-level process. In this method, every pixel is classified as foreground or background. The belief propagation method is a feature-level process.

Figure 1 shows that both algorithms have two major blocks. First, camera motion parameters are estimated. Then, the camera motion parameters are used to detect the moving object. In the following sections, we introduce camera motion estimation first. The details of the two different algorithms are described later.

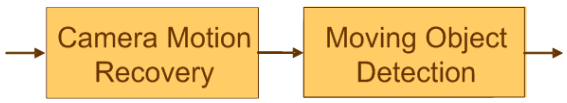


Figure 1: General detection block diagram.

3.1 Related Work

At the earlier development stage of moving-object detection, many methods have been proposed for static cameras. Most of those methods assume that the color or intensity of stationary background may change gradually over time. Some simple

ways can solve this by applying infinite impulse response or a Kalman filter [37, 42] to smooth the color of a background pixel. In order to tolerate the background variation, a better alternative is to use a Gaussian function to describe the distribution of every background pixel [8]. Later, more methods were developed for a variety of background situations. Toyama et al. [64] utilized a linear Wiener filter to learn and predict color changes in every background pixel. But, it is difficult to model the non-periodical background changes. Among those methods, mixture of Gaussians (MoG) [30] is considered a promising method. In MoG, the colors of background pixels are described by multiple Gaussian distributions. MoG performs well on foreground detection in outdoor scenes. However, those methods cannot perform well when the background contains shadows and a wide range of changes, e.g., wavering tree branches and moving escalators. Therefore, more methods have been developed for solving the above issues in detecting moving objects by static cameras. Li et al. [45] chose color feature to describe stationary background objects and color co-occurrence features to represent the moving background object. Based on the information of features, new learning strategies for background changes were proposed to adapt to various changes in the background. KaewTraKulPong et al. [35] utilized different equations at different phases to learn quickly and accurately as well as to adapt effectively to changing environments. A shadow detection scheme is also introduced based on computational color space. These two methods are considered two of the most sophisticated methods.

As cameras become highly pervasive, research on moving cameras has gained more and more attention. For example, Bethke et al. [6] reformulated the estimation problem as a linear function for multiple unmanned aerial vehicles (UAVs) in indoor environments. The algorithm used a simple linear Kalman filter to estimate the object position cooperatively among UAVs. Greenhill et al. [26] proposed an algorithm to generate a panorama for the image sequence captured by a camera in a bus. Gleicher

et al. [24] proposed a method to improve the camera dynamics of casual video.

However, few works relate to detecting moving objects from a moving platform. In moving-object detection, the first intuition is to compensate the camera motion. Homography is an easy approach to compensate camera motion and was extended for moving-object detection in [55, 54]. But, in those cases, the study on camera motion was limited to translation and rotation for a pan/tilt camera. For moderate camera motion, Jung and Sukhatme [34] used a bilinear model to compensate camera motion for a camera mounted on a mobile robot. After compensating the camera motion, an adaptive particle filter was applied to generate a set of weighted particles that estimated the posterior distribution of a moving object. Given those particles, MoG was inferred corresponding to the posterior distribution using an expectation-maximization (EM) algorithm. Similarly, Behrad et al. [5] used an affine transformation to model camera motion. In [5], Behrad et al. focused on the moving vehicle detection and used the aspect ratio of horizontal and vertical line as constraints to verify vehicles. However, for more complicated camera motion, those methods are not applicable. Other than the above methods, geometry properties can also be used for detecting moving objects by a moving camera. Yamaguchi et al. [71] utilized feature points and epipolar lines to detect moving objects. Epipolar geometry is characterized by camera motion and scene structure information. In [71], it assumed that there was no moving obstacle in the initial frame and that the road region in the initial frame was decided according to the height of the camera that was measured when the vehicle is stationery. However, when these assumptions are violated due to the presence of moving obstacles in the initial frame or change of camera height, the application of this method would be restricted. Kang et al. [36] proposed a method that was capable of detecting moving objects using multiview geometric constrains. The relative depth map and epipolar geometry properties were utilized to segment moving objects in this method. However, the approach requires future information which

cannot be known at current time. That makes this method noncausal. A different strategy was presented in [20]. Ess et al. [20] proposed an algorithm for multi-person detection and tracking from a mobile platform. This algorithm performs robustly on people detection by using shape information. However, the algorithm was developed based on a calibrated stereo rig mounted on a moving platform. Our goal is to detect moving objects by using an uncalibrated monocular camera on a moving platform. It is difficult to apply existing methods to effectively subtract image background from videos captured by a gray-level monocular camera in a vehicle. Under normal conditions, the environment is complex. Plane shape model cannot represent the scene. In this situation, using plane shape model will cause incorrect ego-motion estimation or inaccurate current frame prediction. Besides, during foreground detection, there are usually noises caused by model imperfection. The other source of noise is from new appearing background when it is wrongly recognized as foreground. Moreover, only edge portion of moving objects could be detected in each adjacent frame because there is only slight movement when comparing adjacent frames.

3.2 Camera motion estimation

3.2.1 Camera Motion Model

A method for estimating the geometric relationship among pixels across multiple video frames is derived.

The 3D global vector is discussed first. A 3D global vector represents the relation between a point in 3D space and camera motion. The original position is $[X\ Y\ Z]^T$. The camera motion includes 3D translational and rotational movement. After the camera moves, the position becomes $[X'\ Y'\ Z']^T$.

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T = R_Z^{\gamma} R_Y^{\beta} R_X^{\alpha} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + T.$$

By applying every angle for the rotation matrixes and expanding the equation, we have the following equation.

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = \begin{bmatrix} \cos\beta\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \cos\beta\sin\gamma & \sin\alpha\sin\beta\sin\gamma - \cos\alpha\cos\gamma & \cos\alpha\sin\beta\sin\gamma + \sin\alpha\cos\gamma \\ -\sin\beta & \sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}. \quad (45)$$

If angles are small (θ is small.). The approximations of $\sin\theta$ and $\cos\theta$ are θ and one. Also, the higher order terms can be neglected. We can have the following approximation.

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \approx \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}.$$

Then, 3D global vector, $[V_X\ V_Y\ V_Z]^T$, can be obtained.

$$\begin{bmatrix} V_X \\ V_Y \\ V_Z \end{bmatrix} = \begin{bmatrix} X' - X \\ Y' - Y \\ Z' - Z \end{bmatrix} \approx \begin{bmatrix} 0 & -\gamma & \beta \\ \gamma & 0 & -\alpha \\ -\beta & \alpha & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix}.$$

The pixel movement equation can be derived by projecting the 3D global vector onto the image plane. The perspective projection model is shown in Figure 2. The projection of the origin point is $[x\ y]$.

$$x = F\frac{X}{Z},\ y = F\frac{Y}{Z}.$$

After projecting the 3D global vector onto the image plane, the 2D pixel movement, $[u\ v]$, on the image plane can be derived as follows. For a better understanding, α,

β, and γ are replaced by ω_X, ω_Y and ω_Z correspondingly.

$$\begin{cases} \frac{T_Z x - T_X F}{Z} - \omega_Y F + \omega_Z y + \frac{\omega_X}{F} xy - \frac{\omega_Y}{F} x^2 = u(x, y) \\ \frac{T_Z y - T_Y F}{Z} + \omega_X F - \omega_Z x - \frac{\omega_Y}{F} xy + \frac{\omega_X}{F} y^2 = v(x, y), \end{cases} \tag{46}$$

where x, y are known pixel position and u, v are known horizontal and vertical feature point movement. T_X,T_Y and T_Z are translational movement of camera; ω_X,ω_Y, and ω_Z are rotational movement of camera; F is focal length; and Z is real distance between every point and the camera. These camera motion parameters are unknown and need to be estimated.

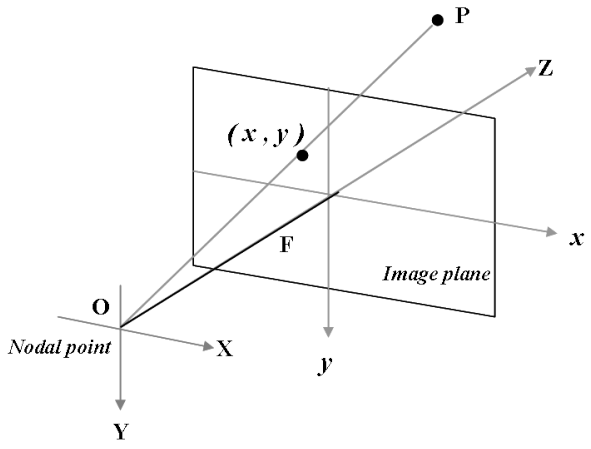


Figure 2: Image plane projection

3.2.2 Estimation Model of Camera Motion Parameters

In this section, an estimation model is proposed. As we can see in Equation (46), Z is in both equations. Without knowing the depth of every pixel, we cannot estimate the camera motion parameters using Equation (46). To solve this problem, a shape assumption of environment is needed. A plane shape is not good enough for a camera in a car because the environment is more complex and keeps changing. Utilizing an ellipsoid to represent the environment is a flexible choice because the ellipsoid shape has three degrees of freedom (length, width, and height) and can adapt to different complex environments.

It is assumed that the camera is at the center of the ellipsoid model and its focal length lies on the Z axis.

$$\frac{X^2}{A^2} + \frac{Y^2}{B^2} + \frac{Z^2}{C^2} = 1 \tag{47}$$

After derivation, we can have the relation between depth, Z, and the position on image plane.

$$\frac{1}{Z} = \sqrt{\frac{x^2}{F^2A^2} + \frac{y^2}{F^2B^2} + \frac{1}{C^2}} \tag{48}$$

In order to do motion recovery, we use Equation (46), (48), and feature motion vectors (u, v) to estimate camera motion parameters. An accurate set of camera motion parameters can be used to calculate the correct quantities of every pixel movement. Then, the current frame can be estimated by the camera motion parameters.

$$(T_Z x - T_X F)\sqrt{\frac{x^2}{F^2A^2} + \frac{y^2}{F^2B^2} + \frac{1}{C^2}} - -\omega_Y F + \omega_Z y + \frac{\omega_X}{F}xy - \frac{\omega_Y}{F}x^2 = u(x, y) \tag{49}$$

$$(T_Z y - T_Y F)\sqrt{\frac{x^2}{F^2A^2} + \frac{y^2}{F^2B^2} + \frac{1}{C^2}} + +\omega_X F - \omega_Z x - \frac{\omega_Y}{F}xy + \frac{\omega_X}{F}y^2 = v(x, y) \tag{50}$$

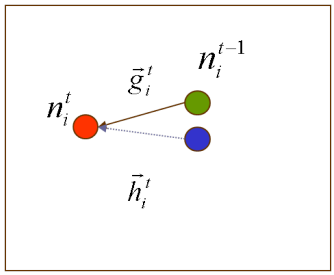


Figure 3: Image plane and feature motion vector

Figure 3 is an image plane. n_i^{t-1} is a feature point on the previous frame and n_i^t is the corresponding feature point in the current frame. $\vec{g}_i^t = (u, v)$ is the feature motion vector of feature point n_i^t. Since we want to estimate camera motion parameters, we generate an initial guess to calculate feature point movement $\vec{h}_i^t$. Ideally, by using the non-linear least squares method, we can get accurate camera motion parameters when $\vec{h}_i^t$ approximates to $\vec{g}_i^t$ with iterations.

We rely on existing feature extraction algorithms for feature detection. The first algorithm is Harris corner detector [27], which is commonly used. The eigen values of

a normal matrix are used for the "cornerness" measure. The Harris corner detector responds accurately to man-made objects with sharp edges and apparent corners, e.g. buildings and vehicles.

Another commonly used feature point detector is the Scale-Invariant Feature Transform (SIFT) points [51]. A keypoint descriptor of this algorithm is created using scale-space extrema detection, keypoint localization and orientation assignment. The SIFT detector is sensitive to complex shaped objects with rich texture.

A faster algorithm of robust feature detector, SURF, was developed by Bay [4]. The standard version of SURF is several times faster than SIFT. SURF is based on the sums of the approximated 2D Haar wavelet responses and makes efficient use of the integral image. As basic image features, it uses a Haar wavelet approximation of the determinant of the Hessian blob detector.

3.2.3 Parameter Estimation

As shown in Equation (49) and (50), solving ego-motion parameters would be complicated. It is difficult to estimate 10 degrees of freedom (3 rotational, 3 translational, focal length and 3 parameters from the ellipsoid model). Plus, the existing square root in the estimation model causes the estimation to be even more difficult. A non-linear least squares method does not consistently provide real solutions and reaches global minimum. However, the other optimization method, genetic algorithm (GA), can provide better solutions for highly complex search spaces and can be easily processed in parallel. Thus, we generate initial guesses first. Next, by inputting the initial guesses, GA is used to update the camera motion parameters to approximate optimization solutions. Then, the GA solutions are provided as initial values to do non-linear least squares optimization. The objective function of GA and the non-linear least squares method is to minimize the sum of squared residuals, r, between

selected feature motion vectors $\vec{g}$ and estimated feature motion vectors $\vec{h}$.

$$S = \sum r^2 = \sum \left\| \vec{g_i^t} - \vec{h_i^t} \right\|^2. \tag{51}$$

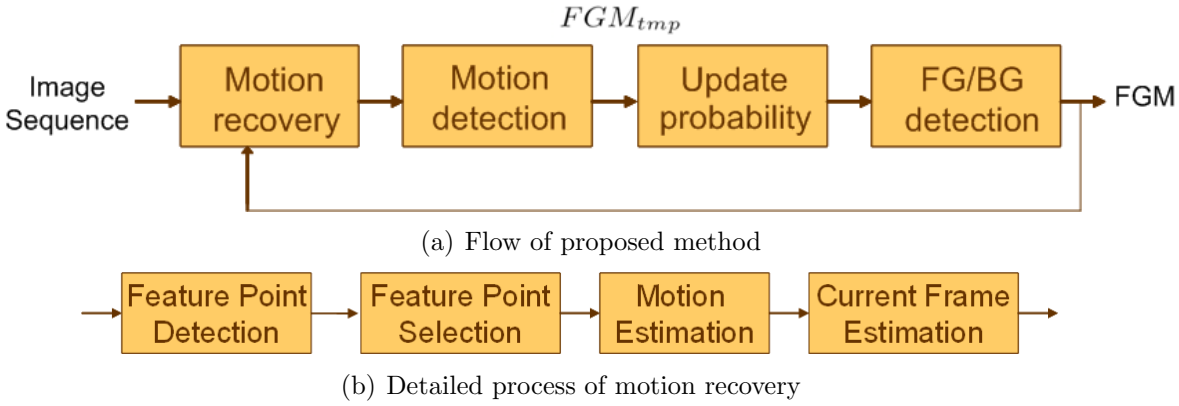
3.3 Bayesian Decision Method

A background/foreground classification method is proposed to effectively segment moving objects from videos captured by a moving camera on a moving platform. Following motion recovery, estimated frame, current frame and feature motion vector are utilized to generate a foreground template. Then, the probabilities update is processed. Based on probabilities, every pixel is classified as either foreground or background.

The process flow of the proposed method is shown in Figure 4(a). The proposed method contains two major parts: motion recovery and segmentation. With the estimation model, motion recovery can be performed. A detailed block diagram of motion recovery is shown in Figure 4(b), The first step of motion recovery is feature point detection. The SURF algorithm [4] is chosen to detect and match feature points in every frame because SURF is robust. The Harris corner detector is not robust enough because the image quality in image sequences is not very good. Although the SIFT algorithm is robust and can provide reliable feature matching, SIFT cannot provide enough matched feature points. Therefore, the SURF algorithm is a better choice for the image sequence shot by a camera from a moving platform. Next, the corresponding feature points between the previous and current frame are matched to obtain feature motion vectors. Only useful feature points are selected for motion estimation. Then, the estimation model utilizes the motion vectors of selected pixels to estimate motion parameters. After estimating ego-motion parameters, the parameters are then used to estimate the current frame.

Following motion recovery, estimated frame, current frame and feature motion

vectors are utilized to a generate foreground template and reduce noise. The foreground template is used to update the pixel-level probability model. Finally, foreground/background detection is processed based on the pixel-level probability model.



(a) Flow of proposed method

(b) Detailed process of motion recovery

Figure 4: Overview of Bayesian decision method

3.3.1 Update Background Information

After motion estimation, camera motion parameters and previous frame, I^{t-1}, can be used to estimate the current frame. Current frame estimation, I^t_{est}, supplies a benchmark for every pixel of current frame, I^t. For current frame estimation, the movement of every pixel is calculated using camera motion parameters. Pixel values of current frame estimation are interpolated by the pixel values of previous frame. The current frame estimation is taken as a reference frame and is compared with current frame. The difference values between I^t_{est} and I^t provide information of foreground and background. However, if we just estimate the current frame from the previous frame, only a small amount of foreground information is obtained. That is because only a small difference exists between I^t_{est} and I^t, and most of differences appear at edges. In order to get better foreground information, we perform interpolation using updated frame, I^{t-1}_{upd}, instead of the previous frame, I^{t-1}. The I^{t-1}_{upd} is updated by the feedback of previous foreground mask FGM^{t-1}. The details of the update process are shown below.

$$I_{upd}^{t-1}(\zeta) = \begin{cases} I_{est}^{t-1}(\zeta),\ \forall\zeta \in \{FGM^{t-1}(\zeta) = 1\} \\ I^{t-1}(\zeta),\ otherwise \end{cases} \tag{52}$$

$$I_{est}^{t} = Interpolate(I_{upd}^{t-1}, \psi) \tag{53}$$

where ζ is pixel position and ψ is a set of camera motion parameters. By repeating this process, it can make accumulative results effectively.

3.3.2 Information Update and Detection

After motion recovery, feature motion vectors and the difference between I^t and I_{est}^t are used to update the probability model for every pixel. The pixel level probability model is used for performing foreground and background classification. However, many sources of noise exist in the difference of I^t and I_{est}^t: interpolation errors, estimation model imperfections, or object movements. We use information obtained from motion recovery to filter out noise sources other than object movements.

3.3.3 Refinement

In this section, feature motion vectors are used to select foreground pixel candidates and delete noise. After deleting noise, the result of the foreground template, FGM_{tmp}^t, is utilized to update the probability model. First, a threshold is applied to delete small interpolation errors, which are a source of noise.

$$I_{diff\_th}^{t} = Threshold_1(I^t - I_{est}^t) \tag{54}$$

Next, for every pixel, when its $I_{diff\_th}^t(\zeta)$ is one, we search feature motion vectors within a window around the pixel. Then the correlation, γ, of $\vec{g_i}$ and $\vec{h_i}$ is calculated for all the feature points within the window. When the expectation of correlation is higher than a threshold, $I_{diff\_th}^t(\zeta)$, of those pixels are treated as noise and are

discarded. The noise is from model imperfection and new appearing background.

$$FGM^t_{tmp}(\zeta) = I^t_{diff\_th}(\zeta) \cdot \{1 - Threshold_2(E[\gamma|\tau, \zeta, \psi])\}, \tag{55}$$

where τ is a set of feature motion vectors. Generally, both noise and feature points appear around edges. Therefore, $E[\gamma]$ is a reasonable choice to provide a margin for ellipsoid model imperfection. Every pixel with $FGM^t_{tmp} = 1$ is taken as a foreground candidate and the probability model is updated based on FGM^t_{tmp}.

3.3.4 Probability Update and Background Detection

A pixel-level probability model and its update procedures are presented in this section. Previous work by Li et al. [45] developed a classification rule for moving object detection by a static camera. We extend their classification rule for moving cameras. By adding camera motion parameters and the different probabilities update method, the following relation is developed for moving-object detection by a moving camera.

$$P(b|\nu_\rho, \zeta, \psi) + P(f|\nu_\rho, \zeta, \psi) = 1 \tag{56}$$

where b is background, f is foreground and ν_ρ is pixel value. By using Bayesian decision rule, the pixel is classified as background if probability values satisfy

$$P(b|\nu_\rho, \zeta, \psi) > P(f|\nu_\rho, \zeta, \psi). \tag{57}$$

Using Bayes rule on the posterior probability of background, it follows

$$P(b|\nu_\rho, \zeta, \psi) = \frac{P(\nu_\rho|b, \zeta, \psi)P(b|\zeta, \psi)}{P(\nu_\rho|\zeta, \psi)}. \tag{58}$$

With these equations, we can obtain the classification rule for moving object detection by a moving camera.

$$Backgound \Rightarrow 2P(\nu_\rho|b, \zeta, \psi)P(b|\zeta, \psi) > P(\nu_\rho|\zeta, \psi) \tag{59}$$

The above three probability sets need to be updated in each pixel. Because every pixel is moving in every frame, predicting probabilities of every pixel by interpolation

is required. The benefit of interpolation is that the probabilities of pixels contain information of adjacent pixels. During classifying the background with (59), it makes the system tolerate small errors caused by model imperfection. To implement Equation (59), we use a quantized histogram to replace probabilities. H^t_ν, $H^t_{\nu b}$ and P^t_b present $P(\nu_\rho|\zeta,\psi)$, $P(\nu_\rho|b,\zeta,\psi)$ and $P(b|\zeta,\psi)$ respectively.

$$H^t_\nu(\rho,\zeta) = \begin{cases} H^t_{\nu\_pre}(\rho,\zeta)+1,\ if\ Q\left[I^t(\zeta)\right]=\rho \\ H^t_{\nu\_pre}(\rho,\zeta),\ otherwise \end{cases} \tag{60}$$

$$H^t_{\nu b}(\rho,\zeta) = \begin{cases} H^t_{\nu b\_pre}(\rho,\zeta)+1,\ if\ FGM^t_{tmp}(\zeta)=0, Q\left[I^t(\zeta)\right]=\rho \\ \alpha\cdot H^t_{\nu b\_pre}(\rho,\zeta),\ if\ FGM^t_{tmp}(\zeta)=1, Q\left[I^t_{est}(\zeta)\right]=\rho \\ H^t_{\nu b\_pre}(\rho,\zeta),\ otherwise \end{cases} \tag{61}$$

$$P^t_b(\zeta) = \begin{cases} \alpha\cdot P^t_{b\_pre}(\zeta),\ if\ FGM^t_{tmp}=1 \\ \alpha\cdot P^t_{b\_pre}(\zeta)+(1-\alpha),\ otherwise \end{cases} \tag{62}$$

$H^t_{\nu\_pre}$, $H^t_{\nu b\_pre}$ and $P^t_{b\_pre}$ are predicted from H^{t-1}_ν, $H^{t-1}_{\nu b}$ and P^{t-1}_b by interpolation. $Q[\cdot]$ is quantization and α is a scale factor. With updated probabilities and the classification rule, background is classified and moving objects can be detected.

3.3.5 Experiments

This section presents experiment results obtained from the proposed method and compares them with results of the existing method using multiview geometric constraints [36], which demonstrated significant detecting results on videos in the presence of strong parallax. The video streams were captured in a forward-moving car by a camera held by a human hand. Because the road is uneven and the human hand is unstable, the captured video streams have a lot of sudden irregular movements. The relative movements between objects and camera are complex and change rapidly.

In these two experiments, $threshold_1$ is 50, $threshold_2$ is 0.7, and search window size is 15. The quantization level of the histogram is 64, and scale factor α is 0.8. γ is the cosine of the angle between the two vectors. In video 1, a pedestrian in front of the camera is moving from right to left. In video 2, a car in front of the camera is moving forward. Figure 5(a)(c)(e)(g) and Figure 5(b)(d)(f)(h) show the experiment results of video 1 and 2, respectively. By applying the proposed method, moving objects in video 1 and 2 are detected and shown in the bounding boxes of Figure 5(a) and 5(b) respectively. Figure 5(c) shows that a pedestrian is accurately detected as foreground in video 1 and Figure 5(d) shows a vehicle is detected in video 2. Figure 5(e) and 5(f) show the corresponding compared results of the method using multi-view geometric constrains. They are the results before applying threshold. As the figures show, no matter what threshold is applied (e.g., Figure 5(g) and 5(h)), there exists more noise in the steady background than in Figure 5(c) and Figure 5(d). As one can see, experiment results show that the proposed method can accurately detect moving objects and also successfully filter out background noise. After moving objects are detected, the moving objects can be accurately tracked using the proposed method in [49].

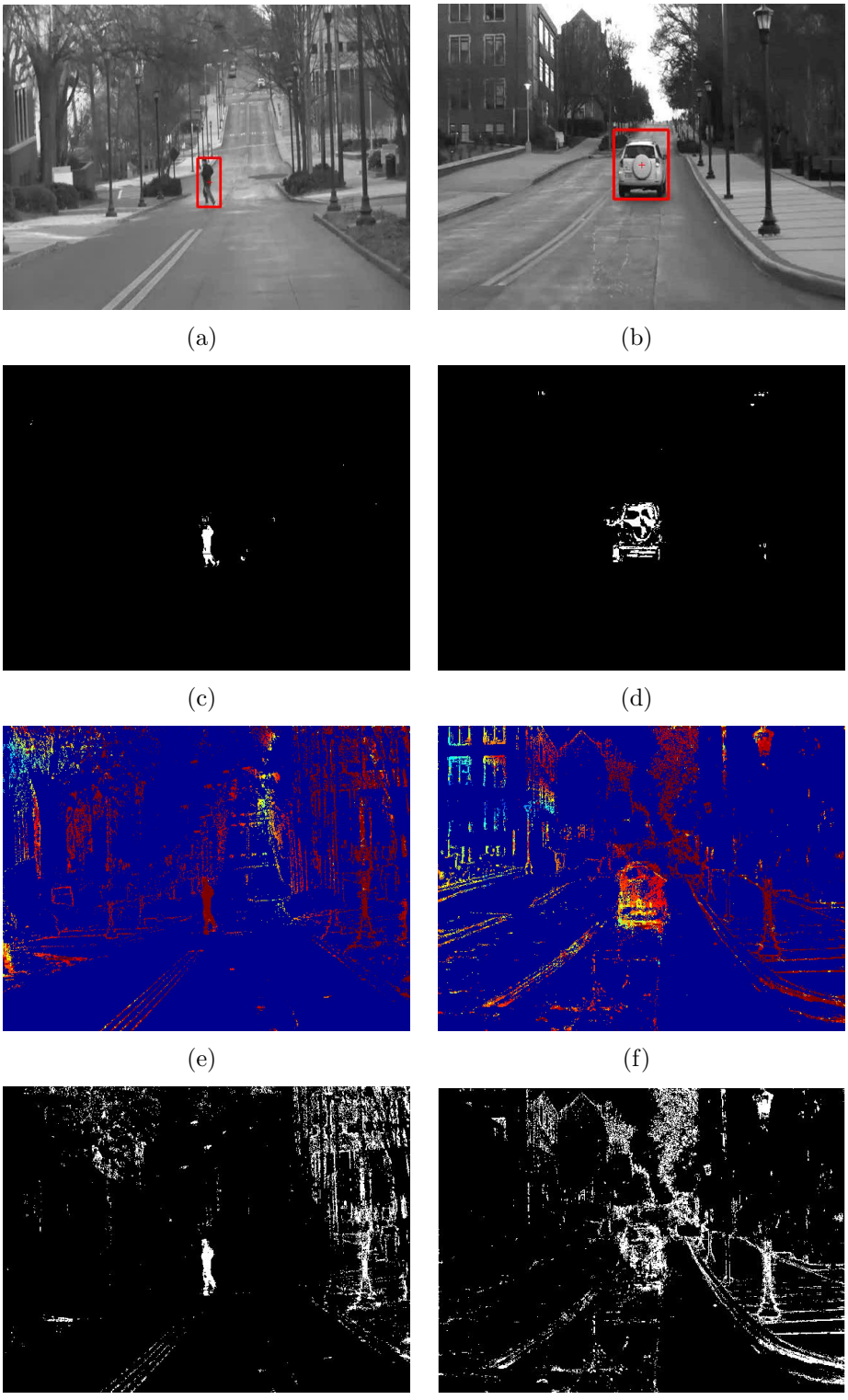


Figure 5: Detecting results of Bayesian decision method

3.4 Belief Propagation Method

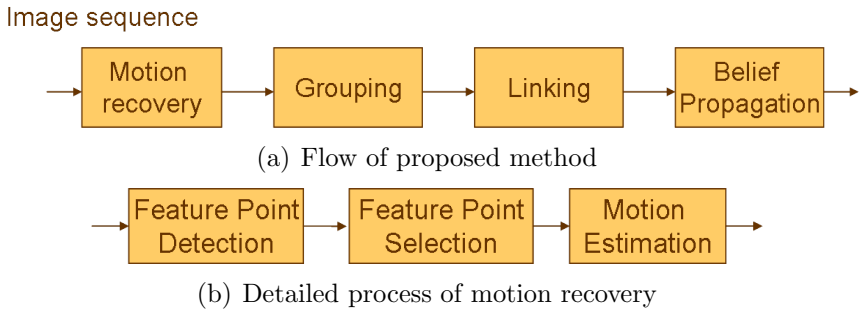
In the Bayesian decision algorithm, pixel-level probability is calculated and updated for every pixel. In this section, we propose another method using less memory and less computational power. We propose a belief propagation method to effectively detect moving objects from videos captured by a camera on a moving platform. In this method, we address the detection problem by creating a graphical model, which uses a belief propagation algorithm.

The process flow of the proposed method is shown in Figure 6(a). The proposed method contains two major parts: motion recovery and moving object detection. A detailed block diagram of motion recovery is shown in Figure 6(b), The first step of motion recovery is detecting feature points. The SURF algorithm is still used in this method to detect and match feature points in every frame. Next, the corresponding feature points between the previous and current frame are matched to obtain feature motion vectors. Only useful feature points are selected for motion estimation. Then, the estimation model uses the motion vectors of selected pixels to estimate motion parameters. In this method, the estimation of current frame is not needed.

In order to classify objects, feature points in each frame are grouped by a hierarchical clustering algorithm after motion recovery. The grouping result of every frame can be used to construct a graphical model by linking related groups in adjacent frames. Finally, a belief propagation algorithm is applied to perform the inference on our graphical model and then moving objects can be detected.

3.4.1 Feature Motion Vector Grouping

With camera motion parameters, one can calculate the motion vector of each feature point $\vec{h_i}$. Because the environment is not a perfect ellipsoid shape, feature motion vectors $\vec{g_i}$ on different parts of the environment may have different deviations with



(a) Flow of proposed method

(b) Detailed process of motion recovery

Figure 6: Overview of belief propagation method

corresponding vector $\vec{h_i}$. But, the deviations are similar around adjacent feature motion vectors. For those adjacent feature motion vectors with similar deviations, they may come from the same part of objects and convey similar information. Thus, we use a clustering algorithm to group similar feature motion vectors. Then, every group contains a single message representing the information of every feature motion vector in its group. There are a number of clustering algorithms. Since the number of groups cannot be known in advance, we cannot use clustering algorithms requiring the information of group number (e.g. k-means). The complexities of some clustering algorithms are high because of the need to calculate density distributions (e.g. mixture of Gaussian and mean-shift). Here, we use the agglomerative hierarchical clustering algorithm [29] to group feature motion vectors in every frame. The agglomerative hierarchical clustering algorithm is a bottom-up process that merges the nearest pair of clusters at each step. The measurement of distance between cluster x_k and $x_{k'}$ is defined as

$$d_{cen}(x_k, x_{k'}) = \|\mu_k - \mu_{k'}\| \tag{63}$$

where μ_k is the center of group x_k. In order to avoid having only one group exist in the final result and merging two groups with large group-to-group distance, every

iteration of clustering is subject to the following constraints:

$$d_{cen} < th_d \tag{64}$$

$$\overline{\theta_k} - \overline{\theta_{k'}} < th_\theta. \tag{65}$$

where $\theta = \angle\left(\vec{g}, \vec{h}\right)$. The clustering iteration will stop until there is no cluster to be merged. Figure 7 shows one of the grouping results.

Figure 7: Grouping result

3.4.2 Linkage

After applying the hierarchical clustering algorithm, every frame has its own grouping result. The grouping results depend on the estimated camera motion parameters and the feature points detected by SURF. Usually the number of groups is different in every frame. And, in different frame, the positions of groups are usually located at different scene spots. We would like to use information of grouping result in previous frame as a reference of current frame, but do not need information of all groups in previous frame. Thus, we need to find which group x_j^{t-1} in the previous frame is potentially related to group x_i^t in the current frame. The i-th group in frame t is denoted as x_i^t. Therefore, we connect groups that are potentially related. By connecting grouping results between adjacent frames, a graphical model can be constructed like that in Figure 8. In order to establish a connection, we need to know

the group center μ_i^t, group standard deviation σ_i^t, and its average feature motion vector $\overrightarrow{\overline{g_i^t}}$ of group x_i^t. To link the group x_i^t in the current frame t to the grouping result of the previous frame, we move the current group position μ_i^t to the previous position by its average motion vector $\overrightarrow{\overline{g_i^t}}$. Then, we link group x_i^t to group x_j^{t-1} in the previous frame $t-1$ when x_j^{t-1} has overlapping area with the previous position of x_i^t. Let l_{ij} indicate the linkage between x_i^t and x_j^{t-1}.

$$l_{ij} = 1,\ if\ \left\|\mu_i^t - \overline{\vec{g}} - \mu_j^{t-1}\right\| - \sigma_i^t - \sigma_j^{t-1} < 0 \tag{66}$$

If there is no overlapping group, we link x_i^t to the nearest group x_k^{t-1} with multiple feature points in the previous frame.

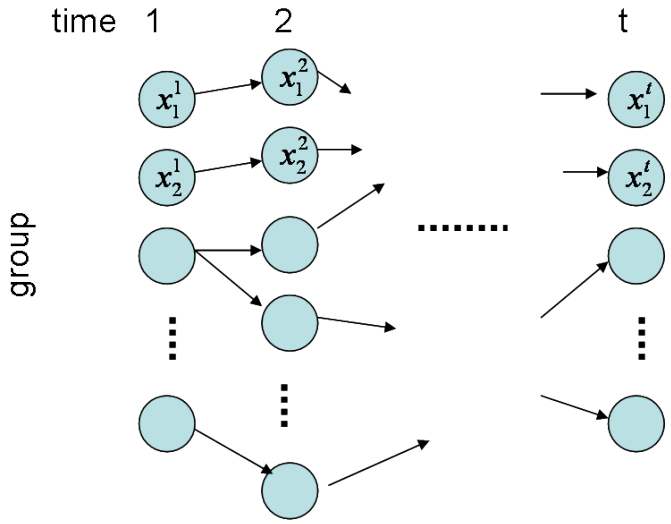


Figure 8: Graphical model

3.4.3 Belief Propagation

After linking groups in adjacent frames, we have a graphical model. Then we use belief propagation (BP) [72] to perform inference on the graphical model. BP is an efficient algorithm based on a message-passing system that stores at each node a separate message for each of that node's neighbors.

Let $m_{ji}(x_i)$ be the message passed from node j to node i. Messages are computed

iteratively using the update algorithm

$$m_{ji}(x_i) = \phi(x_i)\,\psi_{ij}(x_i, x_j) \prod_{k \in N(j)\setminus i} m_{kj}(x_j) \tag{67}$$

where $\psi_{ij}(x_i, x_j)$ is the potential function, and $\phi(x_i)$ is the local evident function. The set of nodes $k \in N(j)$ is termed the neighborhood of node j.

The belief at a node i can be defined by the local evident function $\phi(x_i)$ and the message function $m_{ji}(x_i)$.

$$b_i(x_i) = K\phi(x_i) \prod_{k \in N(i)} m_{ki}(x_i) \tag{68}$$

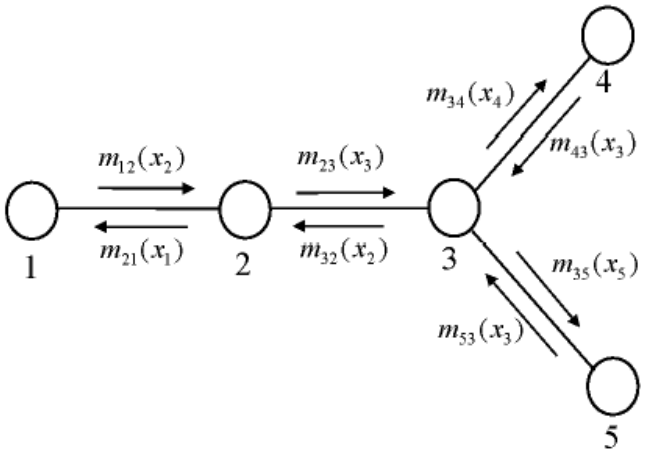


Figure 9: Belief propagation

Figure 9 is a example of belief propagation. For the calculation of the message from node 3 to node 2, it needs to multiply the message from node 4 to node 3 and the message from node 5 to node 3, and then multiply the potential function between node 3 to node 2 and local evident function of node 2.

We apply BP on our graphical model to perform the inference of every group in the current frame using messages passed from groups in the previous frame. The belief of every group x_i^t will be used to detect moving objects. In our graphical model, there is no linkage between groups in the same frame. Messages are transmitted sequentially from groups in frame $t-1$ to groups in frame t. To apply BP on our graphical model,

the message, potential function, and local evident function are express as $m_{ji}^t(x_i^t)$, $\psi_{ij}^t\left(x_i^t, x_j^{t-1}\right)$ and $\phi_i^t(x_i^t)$ respectively. To perform the inference in our implementation, $m_{ji}^t(x_i^t)$, $\psi_{ij}^t\left(x_i^t, x_j^{t-1}\right)$ and $\phi_i^t(x_i^t)$ are defined as follows

$$m_{ji}^t(x_i) = e^{\rho_{ji}^t\left(x_i^t\right)} \tag{69}$$

$$\psi_{ij}^t\left(x_i^t, x_j^{t-1}\right) = e^{\xi_{ij}^t\left(x_i^t, x_j^{t-1}\right)} \tag{70}$$

$$\phi_i^t(x_i) = e^{\zeta_i^t\left(x_i^t\right)} \tag{71}$$

If we take the logarithm of belief function, we have

$$log\left(b^t(x_i)\right) = log(K) + \zeta_i^t\left(x_i^t\right) + \sum_{k \in N(i)} \rho_{ki}^t\left(x_i^t\right) \tag{72}$$

Because $log(K)$ is a constant, which is insignificant, we set $\widetilde{b}^t(x_i)$ as the new belief to be updated.

$$\widetilde{b}^t(x_i) = \zeta_i^t\left(x_i^t\right) + \sum_{k \in N(i)} \rho_{ki}^t\left(x_i^t\right) \tag{73}$$

If we take the logarithm of the message function as well, $\rho_{ij}^t(x_i^t)$ becomes the new message function to be updated iteratively.

$$\rho_{ji}^t(x_i^t) = \zeta_j^{t-1}\left(x_j^{t-1}\right) + \xi_{ij}^t\left(x_i^t, x_j^{t-1}\right) + \sum_{k \in N(j) \setminus i} \rho_{kj}^{t-1}\left(x_j^{t-1}\right) \tag{74}$$

As one can see, the message and belief function become summation formulas. Therefore, ξ_{ij}^t and $\zeta^t(x_i^t)$ need to be designed carefully with normalization factors in order to compare the belief of each group.

3.4.4 Implementation Details

In our implementation, ξ_{ij}^t and $\zeta^t(x_i^t)$ are updated by the following equations:

$$\zeta^t\left(x_i^t\right) = \frac{1}{\eta_i^t}\gamma_i^t\overline{\theta_i^t} \tag{75}$$

$$\xi_{ij}^t\left(x_i^t, x_j^{t-1}\right) = \widetilde{b}_j^{t-1}(x_j)\left(\frac{\Lambda_{ij}^t}{\eta_i^t} - 1\right) \tag{76}$$

$$\Lambda_{ij}^{t} = \alpha \frac{\eta_j^{t-1}}{\left(d_{ij}^{t}\right)^{c}} / \sum_{k} \frac{1}{\left(d_{ik}^{t}\right)^{c}} \tag{77}$$

$$\eta_i^{t} = \sum_{k} \Lambda_{ki}^{t} + \gamma_i^{t} \tag{78}$$

where γ_i^t is the feature number of group i. And, α and c are constants. Λ_{ij}^t is designed for the normalization of the new message function $\rho_{ji}^t(x_i^t)$ and η_j^t is designed for the normalization of new belief function $\widetilde{b}^t(x_i)$. At the initial frame, Λ_{ij}^t is zero, and η_i^0 is γ_i^0. The belief values of groups in each frame are accumulated from previous s frames. Moving objects are detected as groups appearing high belief values consecutively.

3.4.5 Experiments

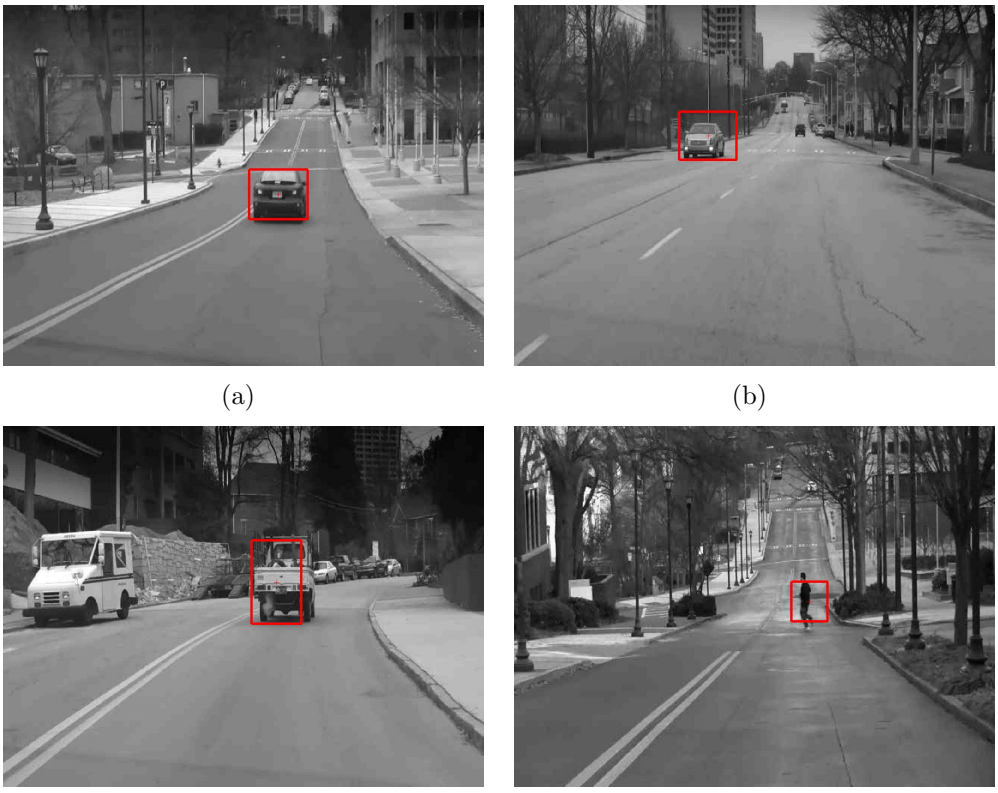
This section presents experiment results obtained from the proposed method. The video streams were captured in a forward-moving car by a camera held by a human hand. The car speed is about 25 MPH. Because the road is uneven and the human hand is unstable, the captured video streams have a lot of sudden irregular movements. The relative movements between objects and camera are complex and change rapidly.

In all experiments, s is 5, c is 2, α is 0.8, th_d is 25 and th_θ is 10. As for the parameters of the genetic algorithm, the number of solutions for each generation is 250 and the number of generations is 750. Mutation probability is 0.0001. Figure 10(a)(b)(c)(d) show results of experiments 1, 2, 3, and 4, respectively. The red box on the moving objects is defined as the feature points of the detected group. In the videos of experiments 1 and 3, a moving car in front of the camera is moving in the same direction. In the video of experiment 3, several static cars are parked at the left side. In the video of experiment 2, a moving car in front of the camera is moving toward the camera. Several cars stop because the traffic light is red. In the video of experiment 4, a pedestrian in front of the camera is moving from right to left.

As the experiment results show, the proposed method can detect moving objects successfully in different environments, even in complex environments in experiments

3 and 4. After the moving objects are detected, they can be accurately tracked using the method proposed in [49].

Compared with the Bayesian decision method, the belief propagation method has less complexity. However, as one can see in the experiments of the belief propagation method, the object position and size are not as precise as the results of the Bayesian decision method. The reason is that the object position and size are defined by matching feature points and the matched feature points are not always uniformly distributed on the moving object.



(a) (b)

(c) (d)

Figure 10: Detecting results of belief propagation method

CHAPTER IV

TRACKING MOVING OBJECTS

Sudden camera or object motion is a typical problem that causes tracking performance to sharply deteriorate. Classical recursive Bayesian estimation is inadequate to track moving objects observed by a rapid-moving and unstable camera since this method cannot resolve the sudden motion problem. We have developed a robust and unconstrained tracking algorithm to overcome the tracking-failure issues. The Markov Chain Monte Carlo (MCMC) technique is adopted to efficiently realize the feature-guided particle filter.

Unlike the methods proposed in [5, 43], our experiment results show that the method for tracking moving objects demonstrates robust performance without the assistance of foreground segmentation and performs accurately in severe tracking environments.

4.1 Related Work

Various algorithms exist for tracking moving objects by static cameras but they are derived from the same basic formulation. The objective is to find the object state characterized by $\{x_k\}_{k=1,2,...}$ based on the observation $\{z_k\}_{k=1,2,...}$. The evolution of x_k is defined as $x_k = f_k(x_{k-1}, m_k)$, and the measurement function is defined as $z_k = g_k(x_k, n_k)$. Usually, f_k and g_k are non-linear and time-varying functions. The noise sequences, $n_k$and m_k, are assumed to be independent and identically distributed (i.i.d.). If f_k and g_k can be modeled as linear functions, a Kalman filter can provide the optimal solution. Boykov and Huttenlocher [9] used a Kalman filter to track vehicles in an adaptive framework. In [9], the object parameters include position and configuration of non-occluded features. At each frame, a maximum a posteriori

(MAP) estimation of the object parameters will be found. However, evolution and observation functions cannot always be modeled as linear functions. If f_k and g_k are non-linear functions,an extended Kalman filter (EKF) or Unscented Kalman Filter (UKF) can be used for optimization. Rosales and Scaroff [59] used the EKF to estimate 3D object trajectories from 2D image motion. And then the trajectory, occlusion and segmentation information are utilized in extracting stabilized views of the moving objects. The most commonly used method is a particle filter, which is based on the stochastic sampling method. A particle filter was first introduced by Isard and Blake [31] in computer vision. The advantage of a particle filter is the ability to handle arbitrary densities. If state space is discrete and the number of states is finite, hidden Markov models (HMM) can be used for tracking. Chen et al. [14] used the HMM formulation for tracking. In [14], a joint data association filter (JPDAF) was used to compute the HMM's transition probabilities, taking into account the intercorrelated neighboring measurement. Besides, multiple hypothesis filter (MHF) provides another way to evaluate the probability of measurement sequence. MHF can be use to track the modes of the state density. Cham and Rehg [11] utilized a variant of MHF for tracking highly articulated objects. To improve efficiency, Comaniciu et al. [17] proposed a mean shift algorithm for nonrigid object tracking. The feature histogram-based target representations are regularized by spatial masking with an isotropic kernel. The masking induces spatially-smooth similarity functions suitable for gradient-based optimization. The target localization problem was formulated using the basin of attraction of the local maxima, and the mean shift procedure was used to perform the optimization. Furthermore, Dore et al. [18] proposed multicue adaptive particle filter-based tracker (MAPT) algorithm to track deformable objects. Shape and color cues were exploited to handle deformable objects.

As for tracking algorithms, the classical particle filter work well for tracking moving objects by steady cameras. But, these methods are not applicable for moving

cameras because they are not robust to global appearance changes and sudden camera motion. Some existing methods for moving cameras are as follows. Meuter et al. [52] used UKF to estimate the movement of people. They assumed that targets and the camera are moving on the same plane and that some camera parameters are known. The moving host and target movements can be modeled as 2D movements on a flat ground-plane. The motion and the measurement model are combined by an UKF. Thus, this method can only be applied to specific environment because it has several constraints. Ess et al. [20] proposed a multi-person tracking algorithm. People were detected using shape information and were tracked by a tracking-by-detection approach. The method integrated stereo depth and visual odometry in the hypothesize-and-test model selection framework. However, this method needs a calibrated stereo rig to obtain depth information and can only track specific trained objects. For tracking algorithms used in moving cameras, the major problem is sudden camera motion because sudden camera motion not only largely changes object position, but also causes image blur. Image blur will cause object appearance changes. Those situations happen frequently, especially for a camera in a car. To solve this kind of problem, the method proposed by Behrad et al. [5] used object detection to re-track the missing target. Kumar et al. [43] proposed an integrated method to overcome the sudden motion problem, but this method made use of both top-down and bottom-up approaches. Both approaches need foreground segmentation information. If foreground segmentation is required to assist tracking, there is less flexibility in embedded computing. Our goal is to develop tracking algorithms using an uncalibrated monocular camera from a moving platform. Moreover, the algorithm is expected to overcome the problem caused by sudden camera motion without using detection information.

4.2 Particle Filter

The state of a target and its observation at time t are denoted as x_t and z_t, respectively. $Z_t = \{z_1, \ldots, z_t\}$ represents the overall observation from time 1 to t. The posterior probability of x_t given observation Z_t at time t can be derived as

$$p(x_t|Z_t) = \alpha \cdot p(z_t|x_t) \cdot \int p(x_t|x_{t-1}) \cdot p(x_{t-1}|Z_{t-1}) dx_{t-1} \tag{79}$$

Equation (79) is a recursive form of Bayesian estimation using prior probability $p(x_{t-1}|Z_{t-1})$ to estimate posterior probability $p(x_t|Z_t)$. The estimation also needs observation model $p(z_t|x_t)$, which is a measurement of z_t given x_t. The state transition model $p(x_t|x_{t-1})$ is used as a motion model to predict current state x_t given previous state x_{t-1}. This formula is valuable in the visual tracking area and has been used in a lot of work.

4.3 Feature-guided Particle Filter

Chang et al. [13] proposed an attractor-guided particle filtering (AGPF) method for articulated hand tracking. This method is a model-based tracking approach that is used to solve tracking difficulties caused by the high degree of freedom (DoF). Chang et al. used prior collected shape attractors $A = \{a_1, \ldots, a_n\}$ to improve the articulated hand tracking performance in [13]. The attractor-guide particle filter is formulated as

$$p(x_t|Z_t, A) \propto p(z_t|x_t) \cdot \int p(x_t|x_{t-1}, A) \cdot p(x_{t-1}|Z_{t-1}, A) dx_{t-1} \tag{80}$$

Particles of the AGPF method are the combination of particles generated by the collected shape model and transition model. The experiment results in [13] show that the method is effective in articulated hand tracking. Jian et al. [32] also utilized the formula for accurate lip contour tracking. Although we do not want to use the collected model to help tracking, the equation is adopted to develop a new estimation model for solving tracking problems of moving cameras. In the proposed method

in this chapter, attractors A in Equation (80) are replaced by observed features F. Therefore, we do not need to collect prior model in our feature-guided particle filtering method. With the careful design of $p(x_t|x_{t-1}, F)$, the performance of tracking by a moving camera will be improved.

4.4 MCMC-based Feature-guided Particle Filter

The state $x = [s_x \ s_y \ v_x \ v_y \ w \ h]$ contains position s, motion vector v, width w, and height h of the target. Our observation likelihood $p(z_t|x_t)$ is defined as being proportional to the l^{th} power of Bhattacharyya coefficient ρ. The observation model is defined as

$$p(z_t|x_t) \propto (\rho(h,k))^l, \tag{81}$$

where h and k are the histograms of the sample state and candidate model correspondingly.

We now consider the transition model. The probability $p(x_t|x_{t-1})$ is modeled by Gaussian distribution,

$$p(x_t|x_{t-1}) \sim N(x_{t-1}, \Sigma_1). \tag{82}$$

where Σ_1 is variance.

The implementation of the integral in the recursive Bayesian estimation Equation (80) is intractable. Usually, the sequential importance sampling (SIS) technique is used to approximate the distribution $p(x_{t-1}|Z_{t-1}, F)$. A set of weighted samples $\{x_{t-1}^k, w_{t-1}^k\}_{k=1}^N$ is generated and used for Monte Carlo approximation for the integral in Equation (80). The posterior can be expressed as

$$p(x_t|Z_t, F) \propto p(z_t|x_t) \cdot \sum_{k=1}^{N} w_{t-1}^k \cdot p(x_t|x_{t-1}^k, F). \tag{83}$$

SIS can draw a set of samples and calculate its weight $\{x_t^k, w_t^k\}_{k=1}^N$ to approximate the posterior $p(x_t|Z_t)$.

However, the Markov Chain Monte Carlo (MCMC) is a more efficient way to achieve the same approximation. We use the MCMC sampling step to replace the SIS step on feature-guided particle filtering. A set of unweighted samples $\{x_{t-1}^k\}_{k=1}^N$ generated by MCMC sampling is used for approximating $p(x_{t-1}|Z_{t-1}, F) \approx \{x_{t-1}^k\}_{k=1}^N$. After obtaining MCMC samples, removing the constant part and applying Monte Carlo approximation, Equation (80) can be expressed as

$$p(x_t|Z_t, F) \propto p(z_t|x_t) \cdot \sum_{k=1}^{N} p(x_t|x_{t-1}^k, F). \tag{84}$$

The Metropolis-Hastings (MH) algorithm is used to generate an unweighted sample set $\{x_t^k\}_{k=1}^N$ with posterior distribution $p(x_t|Z_t)$.

- Transition probability:

  We now consider the feature-guided transition probability $p(x_t|x_{t-1}, F)$. The distribution of $p(x_t|x_{t-1}, F)$ requires careful design to achieve good tracking performance. Because our algorithm is not a model-based approach, the form of $p(x_t|x_{t-1}, F)$ is different from the form in [13]. We design $p(x_t|x_{t-1}, F)$ as a multiplication of $p(x_t|x_{t-1})$ and $p(x_t|F_i)$, where $i = 1, \ldots, n$. x_{t-1} and F are assumed independent. The feature-guided transition probability has the following expression:

  $$p(x_t|x_{t-1}, F) = p(x_t|x_{t-1}) \cdot \prod_{i=1}^{n} p(x_t|F_i) \tag{85}$$

  By using this design of $p(x_t|x_{t-1}, F)$, the posterior probability $p(x_t|Z_t)$ would be transformed to a more accurate distribution by the observed features. Substituting Equation (85) into Equation (84), we get

  $$p(x_t|Z_t, F) \propto p(z_t|x_t) \cdot \left[\prod_{i=1}^{n} p(x_t|F_i)\right] \cdot \left[\sum_{k=1}^{N} p(x_t|x_{t-1}^k)\right] \tag{86}$$

  where the probability $p(x_t|x_{t-1})$ is the probability formed by the motion model and $p(x_t|F)$ is the probability formed by the observed features in every frame.

- Acceptance ratio:

  In the MH algorithm, a proposed move is generated by the proposal distribution $q(x', x)$. The move is accepted with an acceptance ratio a where

$$a = min\left\{1, \frac{\pi(x')q(x', x)}{\pi(x)q(x, x')}\right\}. \tag{87}$$

  If rejected, the move x' is discarded and x remains unchanged. In this way, distribution of samples generated by MCMC will approximate desired distribution π. In this paper, the desired distribution is defined as

$$\pi(x) = p(z_t|x_t) \cdot \left[\prod_{i=1}^{n} p(x_t|F_i)\right] \cdot \left[\sum_{k=1}^{N} p(x_t|x_{t-1}^k)\right]. \tag{88}$$

Algorithm 1 MCMC-based feature-guided particle filtering

Initialization: generate the unweighted particle set $\{x_1^k\}_{k=1}^N$
while $t > 1$ **do**
  for $k = 0$ to MN **do**
    MCMC sampling
    a) sample x' from proposal density $q(x', x)$
    b) compute the acceptance ratio a by Equation (87)
    c) accept x' with acceptance ratio; if reject, x remains unchanged.
  end for
  Obtain new sample set $\{x_t^k\}_{k=1}^N$ from every M sample in MCMC sample pool
  $t \leftarrow t + 1$:
end while

4.5 Observed Features

The observed features for the feature-guided particle filter are demonstrated in this section. The observed features are found by the Harris corner detector in every frame. Corresponding feature points are matched by correlation [75] between every adjacent frame. To filter out matched-point errors quickly, we use RANSAC [22] to select feature points from the same plane. Our environment model is complex and feature points of the background are not in the same plane. We need to select feature points from another plane and combine them. The combination result can generally rule

out the feature points of moving objects. The feature points before selection and after selection are shown in Figure 11. As one can see in Figure 11(a) and 11(b), most feature points on the moving objects are not selected. Therefore, we have two matched feature point sets and their corresponding motion vector sets to be used (before RANSAC selection and after RANSAC selection).

We now describe the two feature probability models used in our method. The first feature probability model $p(x_t|F_1)$ uses the feature motion vector to conduct the position of state. The motion vector of proposed sample x is compared to all the feature motion vectors in the area of x. The area of x is defined by the position and size component of the state x. Feature motion vectors are modeled by Gaussian distribution as well. The second feature probability model $p(x_t|F_2)$ uses the feature point ratio to adjust the size of the state. The first feature model

$$p(x_t|F_1^j) \sim N(F_1^j, \Sigma_2), \tag{89}$$

where $j = 1, \ldots, c$, F_1^j is the feature motion vector in the area of x,

$$p(x_t|F_1) = \frac{1}{c} \cdot \sum_{j=1}^{c} p(x_t|F_1^j). \tag{90}$$

The second feature probability model

$$p(x_t|F_2) = (c - d)/c \tag{91}$$

is feature point ratio, where c is the number of feature points (before selection) in the area of x and d is the number of RANSAC feature points (after selection) in the area of x. If no feature point is in the area of x ($c = 0$), $p(x_t|F_2)$ is set to one.

4.6 Experiments

Experiment results are presented in this section and show that this method can track objects accurately even in severe and complex conditions. The video streams were captured in a car by a camera held by a human hand. Because the road is uneven and

(a) (b)

Figure 11: Feature points (a) before selection, (b) after selection.

the human hand is unstable, the captured video streams have a lot of sudden irregular movements. The relative movements between objects and camera are complex and change rapidly.

Scene 1 is as follows. The observing camera is moving forward in a car. A moving car in front of the camera is moving in the same direction, and a person is moving toward the camera. Several static cars are parked at the left side.

Scene 2 is as follows. The observing camera is moving forward. A car in front of the camera is moving from left to right.

One hundred particles are used and the length of the thinning interval is 5 ($N = 100$, $M = 5$). We model the proposal distribution $q(x', x')$ by a Gaussian distribution. All variances are set to 3. Figure 12 shows an example of a failure mode by classical particle filtering. The inconsistency of camera movement often causes tracking failure. This figure also shows that the experiment environment is severe. Figure 13 shows accurate tracking of a vehicle by the MCMC-based feature-guided filtering. In this experiment, only the first feature model is used. Tracking is successful but there are biases in the target position, especially target size. Figure 14 shows unstable tracking while only the second feature model is used. Although the tracking performance is superior over a classical particle filter, the tracker misses the target in this experiment. Figure 13 and 14 show that using only one of the feature models is not robust enough.

(a) (b) (c)

Figure 12: Classical particle filter (a) frame 1, (b) frame 3, (c) frame 7.

(a) (b) (c)

Figure 13: MCMC-based feature-guided particle filter with first feature model only (a) frame 1, (b) frame 22, (c) frame 47.

Figure 15 shows robust tracking using both feature models. The position and size of the target have only small biases. Figure 16 shows another example of robust tracking using MCMC-based feature-guided filtering in a different environment.

Figure 17 shows a tracking example with sudden camera motion. Because of this motion, the target position moves a lot, and the image shown in Figure 17(b) is blurred. Tracking using both feature models is still successful in this condition. This figure demonstrates that the proposed method has a strong tracking ability to overcome the tracking difficulties caused by sudden camera motion and significant target appearance changes.

(a) (b) (c)

Figure 14: MCMC-based feature-guided particle filter with second feature models only (a) frame 1, (b) frame 22, (c) frame 47.

(a) (b) (c)

Figure 15: MCMC-based feature-guided particle filter with 2 feature models (a) frame 1, (b) frame 22, (c) frame 47.

(a) (b) (c)

Figure 16: MCMC-based feature-guided particle filter with 2 feature models (different environment) (a) frame 1, (b) frame 5, (c) frame 15.

(a) (b)

Figure 17: Accurate tracking when significant target appearance change and sudden camera motion exist. (a) frame 10, (b) frame 11.

CHAPTER V

DYNAMIC SCENE ANALYSIS FOR ON-ROAD OBJECT DETECTION AND TRACKING

In recent work, researchers have successfully built the object classifier for object detection. Most approaches operate with a pre-defined class and require a model to be trained in advance. Such work remains trapped in a "closed universe" recognition paradigm, but a much more exciting paradigm of "open universe" datasets promises to become dominant in the very near future [50, 63].

Some approaches of scene analysis from a moving vehicle require multi-viewpoint, multi-category object detection [44]. Those approaches use 3D depth information as a reference to analyze the scene. However, multiple cameras may not always be available for a vehicle. There is a need for an approach that uses a monocular camera. In addition, the scene from a moving vehicle is changing all the time and has a wide variety of objects. Thus, off-line training methods on those applications are somewhat limited. An analysis method that does not need off-line training provides a good alternative in case the trained classifier failed to detect objects

To overcome such problems and generate effective results without the above-mentioned restrictions, we present two systems with novel approaches for multi-object detection and tracking using a monocular camera on a moving platform without knowing the camera's intrinsic parameters or camera motion parameters. These analysis methods are developed for on-line detecting the road region and on-road objects: one based on feature-point analysis and the other on superpixel analysis.

5.1 Related Work

While a great deal of research effort has been dedicated to detection algorithms, it is difficult to generally apply existing methods to detect vehicles from videos captured by a mono camera on a moving platform. Yamaguchi et al. [71] proposed a road region detection method by estimating the 3D position of feature points on the road. The feature points and epipolar lines are utilized to detect moving objects. This method assumes that there is no moving obstacle in the initial frame and that the road region in the initial frame is decided according to the height of the camera that is measured when the vehicle is stationery. However, when these assumptions are violated, the application of this method would be restricted due to the presence of moving obstacles in the initial frame or a change of camera height. Kang et al. [36] use multiview geometric constraints to detect objects. However, the approach is non-causal since future information is required in this approach. Ess et al.[20] developed a robust algorithm for detecting and tracking pedestrians from a mobile platform. However, this algorithm was developed for a stereo rig, and the calibration of the stereo rig is required in order to use depth information in this algorithm. Wojek et al. [69] proposed a method to perform 3D scene modeling and inference using a monocular camera in a car. This method uses the trained features to label the road and sky, and to detect objects in the scene. Leibe et al. [44] estimate Structure-from-Motion (SfM) and scene geometry with stereo rig. Then, multiple trained models are fused to obtain 3D localization and trajectory. The classifiers used in those methods need to be trained off-line. One of the main disadvantages of off-line training method is the need to collect and train data in advance for a specific application.

5.2 Feature point Analysis

An on-line learning method is proposed for detecting the road region and on-road objects by analyzing the videos captured by a monocular camera on a moving platform.

Most existing methods for moving-camera detection impose serious constraints or require offline learning. In this approach, the key feature points of the road region are learned based on the detected and matched feature points between adjacent frames without using camera intrinsic parameters or camera motion parameters.

The process flow of the proposed method is shown in Figure 18. The proposed method contains four parts: key feature point learning, feature point classification, road region labeling, and object detection. First, feature point detection and matching are performed in adjacent frames. A probability model based on the Bayesian rule is proposed to learn the key feature points. The key feature points are then used to classify the rest of the feature points applying the conditional probability model. Then, road region boundaries are defined using the classified feature points. Based on the detected road region, the objects on the road are detected by exploiting the outliers of the feature points on the road.

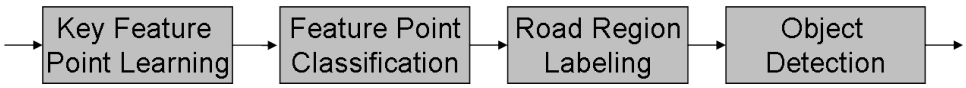


Figure 18: Flow of proposed method.

5.2.1 Key Feature Points Learning

In order to develop a probability model to perform key feature point learning, the characteristics of the road region are considered. In general, fewer feature points can be detected on the road region because the road region is flat and has less texture. And, fewer feature points are matched on the road region. In other words, the similarity of road regions causes a higher rate of mismatching the feature points on the road region. As a result, the matched feature motion vectors have less angle regularity on the road region. Therefore, we use the density of matched feature points and angle regularity of matched feature points to learn the characteristics of the feature points on the road.

Based on the Bayesian rule, the posterior distribution for the scene state X given image evidence η in terms of a prior $P(X|\varsigma)$ and an observation model $P(\eta|X,\varsigma)$ is defined as

$$P(X|\eta,\varsigma) \propto P(X|\varsigma) \cdot P(\eta|X,\varsigma), \tag{92}$$

where ς is pixel position. The scene state X consists of the states of the road region.

The goal of this work is to infer the state X from video captured by a monocular, forward-facing camera in a car. The camera is uncalibrated, and the camera motion parameters are unknown. Meanwhile, we avoid estimating the background structure of the scene. Without knowing any intrinsic and extrinsic parameters, the algorithm is developed using the characteristics of the feature points.

Because the camera is forward-facing, the probability of the road region $P(X|\varsigma)$ can be assumed to follow a normal distribution with mean at the bottom of the image:

$$P(X|\varsigma) \propto N(V;\mu_V,\sigma_V), \tag{93}$$

where V is the vertical position of ς.

The observation model $P(\eta|X,\varsigma)$ fuses the feature density and angle regularity properties of the matched feature points:

$$P(\eta|X,\varsigma) = \psi(d|\varsigma) \cdot \psi(\omega|\varsigma). \tag{94}$$

The feature density potential $\psi(d|\varsigma)$ models the density of matched feature points given the pixel position ς. The feature density potential is defined as

$$\psi(d|\varsigma) = e^{\bar{\kappa}_\varsigma}, \tag{95}$$

where $\bar{\kappa}_\varsigma$ is the number of matched feature points within the window W_ς with size w_s and postion ς.

The angle regularity potential $\psi(\omega|\varsigma)$ describes how well the matched feature points around pixel position ς satisfy the angle regularity. The angle regularity potential is defined as

$$\psi(\omega|\varsigma) = e^{-\Delta\theta} = e^{-\sum_{n_i \in W_\varsigma} |\theta_{n_i} - \bar{\theta}_\varsigma|}, \tag{96}$$

where $\bar{\theta}_\varsigma$ is the average angle of feature motion vecters within the window W_ς.

The inference probability can be defined as

$$\tilde{P}(X|\eta,\varsigma) = P(X|\varsigma) \cdot P(\eta|X,\varsigma). \tag{97}$$

$\hat{P}(X|\eta,\varsigma)$ is the normalized form of $log(\tilde{P}(X|\eta,\varsigma))$, and is used to learn the key feature points:

$$\hat{P}(X|\eta,\varsigma) = \frac{log(\tilde{P}(X|\eta,\varsigma)) - min(log(\tilde{P}(X|\eta,\varsigma)))}{max(log(\tilde{P}(X|\eta,\varsigma))) - min(log(\tilde{P}(X|\eta,\varsigma)))}. \tag{98}$$

The key feature points τ_i are defined as the matched feature points with $\hat{P}(\eta|X,\varsigma)$ smaller than the threshold T_k:

$$\tau = \{n_j : \ \hat{P}(X|\eta,n_j) \ < \ T_k, \ \forall \ j\}, \tag{99}$$

where n_j is the j^{th} detected feature point.

5.2.2 Feature Point Classification

After the key feature points τ are learned, the characteristics of the key feature points are exploited to classify the rest of the feature points. In this approach, a cascade framework is adopted to classify the feature points as in [67]. A cascade classifier can increase detecting performance and radically reduce computational time. In offline training methods, classifiers are trained with annotated data using techniques like SVM or AdaBoost etc. Those techniques are not appropriate in our case because we do not have annotated data in this method. We classify feature points using a particle filter. Every feature of the key feature points is treated as an equally weighted particle in the probability model. Our cascade classifier is shown in Figure 19. Two classifers are cascaded: one uses the coefficients of the Walsh-Hadamard transform, and the other uses the coeficients of the Haar Wavelet transform. The popular HOG feature is not used as our classifier because the road region does not have a rich texture. The coefficients of the Walsh-Hadamard transform (WHT) [2] and the diagonal, horizontal

and vertical coefficients of the Haar Wavelet transform (HWT) [57] are computed as the features for classification. For classification purpose, the conditional inference probability models are applied to infer the likelihood between feature points.

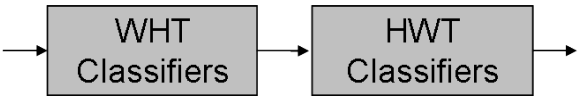


Figure 19: Cascade Classifier

The logarithm of the conditional inference probability for the WHT feature is defined as

$$-log\left(P_{WH}(r|n_j,\{\tau_i\})\right) = \sum_i \left\|f_{WH}(\tau_i) - f_{WH}(n_j)\right\|_1, \tag{100}$$

where $f_{WH}(\tau_i)$ is the WHT feature at the position τ_i and $\|\cdot\|_1$ is 1-norm.

The logarithm of the conditional inference probability model for the HWT feature is defined as

$$-log\left(P_{HW}(r|n_j,\{\tau_i\})\right) = \sum_i \left\|f_{HW}(\tau_i) - f_{HW}(n_j)\right\|_1, \tag{101}$$

where $f_{HW}(\tau_i)$ is the HWT feature at the position τ_i.

In the first classifier, the logarithms of the conditional inference probabilities are used to classify the detected feature points. γ_{WH} is the set of feature points that are classified as the feature points on the road using WHT features:

$$\gamma_{WH} = \{n_j :\ -log\left(P_{WH}(r|n_j,\{\tau_i\})\right)\ <\ T_{WH},\ \forall\ j\}. \tag{102}$$

In the second classifier, the outputs of the first classifier are further classified using HWT features:

$$\gamma = \{\gamma^j_{WH} :\ -log\left(P_{HW}(r|\gamma^j_{WH},\{\tau_i\})\right)\ <\ T_{HW},\ \forall\ j\}, \tag{103}$$

where γ is the set of feature points that are classified as the feature points on the road.

WHT features are computed from the first 16 coefficients of the Walsh-Hadamard transform. This transform is a discrete approximation of the cosine transform and can be computed efficiently. Before the WHT features are calculated, the input image is normalized with zero mean and unit variance.

Haar Wavelets have been introduced by Papageorgiou and Poggio [57] for people detection. The diagonal, horizontal and vertical coefficients of the Haar Wavelet transform are used as the HWT features. HWT features are computed from the absolute responses of horizontal, vertical, and diagonal wavelet types.

5.2.3 Road Region Decision

After the feature points on the road γ are classified, the classified points are used to define the boundaries of the road region. Then, a road-labeled map can be generated by the boundaries of the road region.

In this approach, we focus on the application of the front-facing cameras in a car. The road region on the image plane is non-increasing from bottom to top. An algorithm is developed to define the boundaries of the road region. The car is moving forward; therefore, the region closer to bottom of the image plane has a higher probability of being road. First, the boundaries of the road region are decided from bottom to top order. During the first k steps, the boundaries are purely decided by the region of feature points on the road. After the first k steps, the boundaries of the road region are decided with consideration of previous boundaries. The objects will affect the decision of road boundaries, if they are on the road. The following procedures are designed to prevent the feature points on the objects from affecting the boundary decision. If the boundaries shrink too much, we search feature points in γ_{WH} within the previous boundary plus a margin. If there is no feature point within that region, the boundary is set by a portion of the previous shrinking rate.

In Algorithm 2, LB_j and RB_j represent the left boundary and right boundary at

Algorithm 2 Road Boundary Decision

first k steps: road region are decided by the classified feature points $\{\gamma\}$
for $j = k + 1$ to $height/step$ **do**
  search $\{\gamma_i\}_j$
  $LB_j = min(\{\gamma_i^{(x)}\})$, $RB_j = max(\{\gamma_i^{(x)}\})$
  if $LB_j - LB_{j-1} > th_b$ **then**
    if search $\gamma_{WH,i}^{(x)}$ from $LB_{j+1} - m$ to $LB_{j-1} + m$ **then**
      $LB_j = \gamma_{WH,i}^{(x)}$
    else
      $LB_j = LB_{j-1} + \alpha \cdot (LB_{j-1} - LB_{j-2})$
    end if
  end if
  if $RB_{j-1} - RB_j > th_b$ **then**
    if search $\gamma_{WH,i}^{(x)}$ from $RB_{j-1} + m$ to $RB_{j-1} - m$ **then**
      $RB_j = \gamma_{WH,i}^{(x)}$
    else
      $RB_j = RB_{j-1} - \alpha \cdot (LB_{j-2} - LB_{j-1})$
    end if
  end if
  if no γ_i is found within $[LB_{j-1} - m,\ RB_{j-1} + m]$ **then**
    $LB_j = LB_{j-1}$, $RB_j = RB_{j-1}$, break
  end if
end for

j^{th} step. m is a margin. α is a positive scalar small than one. $\gamma_{WH,i}^{(x)}$ is the horizontal position of feature point $\gamma_{WH,i}$. $\{\gamma_i\}_j$ is defined as:

$$\{\gamma_i\}_j = \{\gamma_i : \gamma_i^{(x)} \in [LB_{j-1} - m,\ RB_{j-1} + m]\,, \forall\ i\}. \tag{104}$$

After all boundaries are defined based on algorithm 2, the boundaries are smoothed.

5.2.4 Object Detection

After the road region is defined, the objects on the road can be identified by the outliers of the feature points on the road region. But, the feature points on the road stripes are outliers as well. A filtering method can be used to detect the outliers on objects or on the road stripes. We apply a 2D rectangle filter on outliers map, the inliers map, and road region map. The output of the filtering results is used to filter out the outlier feature points on the road stripes. The filtered feature point outliers are clustered. Cluster smaller than threshold T_s are discarded. The hierarchical clustering method in [46] is then used to group feature points on the objects.

5.2.5 Experiments

This section presents experiment results obtained from the proposed method. The video streams were captured by a camera in a forward-moving car with the camera held by a human hand. The car speed is about 10 to 35 MPH. The videos are recorded at a frame rate of 10Hz and a resolution of 640x480 pixels. Because the road is uneven and the human hand is unstable, the captured video streams have a lot of sudden irregular movements. The relative movements between objects and the camera are complex and change rapidly. In these experiments, T_k is 0.5, T_{WH} is 0.75, T_{HW} is 2, m is 60, step is 20, w_s is 60, and α is 0.7. The feature points are detected and matched by SURF algorithm [4].

Figure 20 (a) shows the results of matched feature points. As one can see, the road region has fewer matched feature points and a higher mismatching rate because the

road region is flat and has less texture. This figure demonstrates the characteristics of the feature points on the road that we use to develop the learning algorithm. These matched feature points are then used to calculated the inference $\hat{P}(X|\eta, \varsigma)$. Figure 20 (b) shows the distribution of $\hat{P}(X|\eta, \varsigma)$.

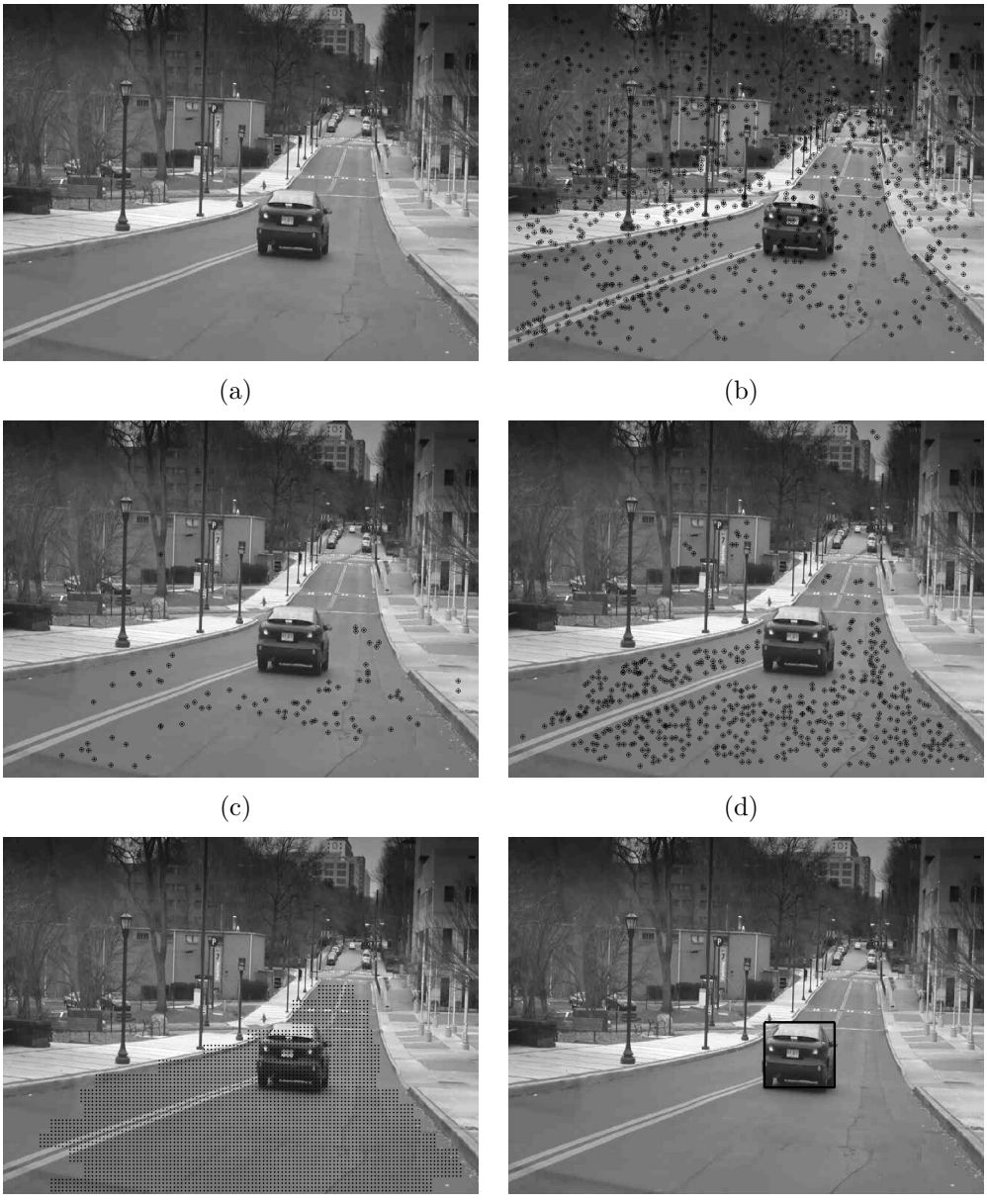
(a) (b)

Figure 20: Experiment (a) matched feature points , (b) the distribution of $\hat{P}(X|\eta, \varsigma)$.

In the video of experiment 1, a car in front of the camera is moving forward. In the video of experiment 2, two cars in front of the camera are moving forward. In the video of experiment 3, a car is moving forward and another car is moving toward the camera. Figures 21, 22, and 23 show the results of experiment 1, 2 and 3 respectively. The original images are shown in Figure 21(a), 22(a), and 23(a). Figure 21(b), 22(b), and 23(b) show the matched feature points. The black-starred feature points are detected and matched feature points using the SURF algorithm. The black-starred feature points in Figures 21(c), 22(c), and 23(c) show the learned key feature points τ. Our learning process can provide more reliable and representable feature points for classification. Therefore, as one can see, the numbers of the learned key feature points are relatively small compared to the number of feature points on the road region. The learned key feature points τ are used to classify the rest of the detected feature points n_i. Figures 21(d), 22(d), and 23(d) show the classified feature points. The black-starred feature points are the feature points classified as the points on

the road γ. In these figures, most feature points on the road are classified correctly. Feature points on the objects and some feature points on the road mark are classified as outliers. Figures 21(e), 22(e), and 23(e) show the results of the detected road region. The road region is marked with black dots. The road region is defined by the classified feature points. As one can see, although there are some classified feature points not on the road, the road region still can be decided correctly. Detected objects are shown in Figures 21(f), 22(f), and 23(f). These figures show that the feature points on the road mark are filtered out successfully, and objects are detected.

As the experiment results show, the proposed method can successfully detect single or multiple objects on the road. In addition, no matter whether the objects are moving forward or moving toward the camera, the proposed method is able to perform significant detecting results.



(a) (b)

(c) (d)

(e) (f)

Figure 21: Experiment 1 (a) original image, (b) matched feature points, (c) key feature points, (d) feature points on the road region, (e) road region, (f) detection.

Figure 22: Experiment 2 (a) original image, (b) matched feature points, (c) key feature points, (d) feature points on the road region, (e) road region, (f) detection.

Figure 23: Experiment 3 (a) original image, (b) matched feature points, (c) key feature points, (d) feature points on the road region, (e) road region, (f) detection.

5.3 Superpixel Analysis

The task we address in this section is dynamic scene analysis for detecting and tracking multiple vehicles from a moving, camera-equipped platform. We seek to detect other traffic participants in the environment. Such capability has obvious applications in driving-assistance systems.

In this approach, bottom-up segmentation is applied on the input images to first get the superpixels. The segmentation results provide regional information and make the analysis more efficient than pixel-level analysis. The scene is then parsed into less segmented regions by grouping similar superpixels and the parsing result is used to estimate the road region. Next, a classification process is performed to detect the outlier of superpixels on the road region. Superpixel outliers and the lines detected in the scene are used to detect vehicles on the road. Finally, tracking is achieved and could be used to feed back to further guide vehicle detection in future frames.

The section is structured as follows. The superpixel merging method is presented in Section 5.3.1. Section 5.3.2 describes the approach to classifying superpixels on the road. Section 5.3.3 presents the multi-vehicle detection method. A tracking method is explained in Section 5.3.5. Section 5.3.6 presents experimental results.

5.3.1 Merging

We want to analyze the video streams based on the content of the images. Analyzing video streams using superpixels is more efficient than pixel-based analysis. Superpixels that are generated by bottom-up segmentation can provide spatial information for aggregating pixels that could belong to a single object and reduce analysis complexity. Usually, some segmentation methods, like those in [21, 53], generate over segmented results. Segmentations similar and belonging to the same objects are separated by its own or other objects' edges. To achieve efficiency, the segmentations of the same objects should be merged together. Here, we propose a method to merge similar

segmentations that belong to the same vehicle. The likelihood $L_g(S_i, c_j)$ is presented to evaluate the similarity between superpixel S_i and the superpixel group c_j:

$$L_g(S_i, c_j) = \omega_1 L_c(S_i, S_j) + \omega_2 L_f(S_i, S_j) \ \forall S_i \in \vartheta_{c_j}, \tag{105}$$

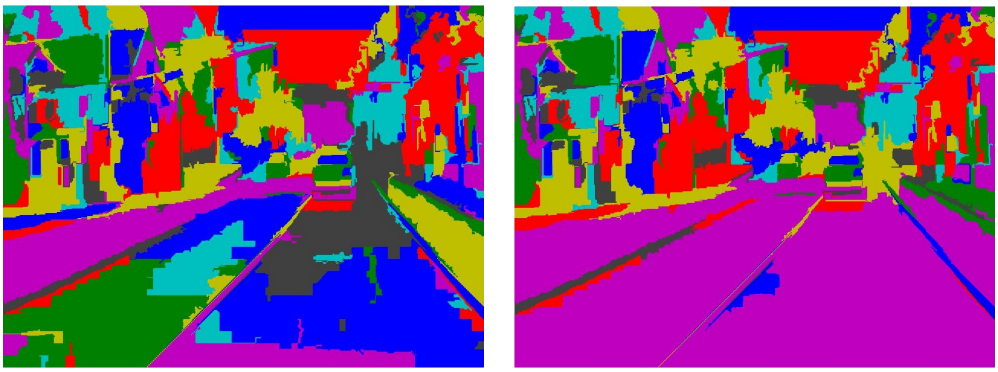
where $L_c(.)$ is the color likelihood, $L_f(.)$ is the feature likelihood, S_i is one of the neighbor superpixels of group c_j, S_j is the initial superpixel in group c_j, ϑ_{c_j} is the set of neighbor superpixels of group c_j, and ω_i is the weighting. When computing the likelihood, superpixel S_j is used to compare with superpixel S_i. The merging process for c_j is initiated from the largest superpixel and performed in order of descending size. Simply using RGB image model cannot deal with shadow problems. Therefore, the HSV image model is also used in the color likelihood to help the merging process. Means and variances of the RGB and HSV values in superpixels are calculated. The means and variances are taken as Gaussian random numbers. The logarithm of the probability $P_c(S_i|S_j)$ is used as the color likelihood $L_c(S_i, S_j)$. For the feature likelihood $L_f(.)$, the feature used here is the coefficients of the Walsh-Hadamard (WH) transform, and the size of the WH transform is determined by the superpixel's size. The logarithm of the conditional inference probability for the WH feature is used as the feature likelihood and is defined as follows:

$$-log\left(P_{WH}(S_i|S_j)\right) = \sum_i \left\| f_{WH}(S_i) - f_{WH}(S_j) \right\|_1, \tag{106}$$

where $f_{WH}(S_i)$ is the mean of WH feature in superpixel S_i and $\|\cdot\|_1$ is 1-norm. The popular histogram of oriented gradients (HOG) feature is not utilized to compare the similarity of superpixels for merging because the road region does not have rich texture. The merging process is recursive. After every round of merging, the new neighbors of c_j are inspected until no more merging can be done. Since superpixels are bounded by edges, the superpixels are expanded for multiple pixels by morphological operation in order to find similar neighbors. When the likelihood $L_g(S_i, c_j)$ is high, we

merge the superpixel S_i into group c_j. After this merging process, similar neighboring superpixels are grouped together.

Figure 24(a) shows the segmentation results of superpixels. As one can see, the segmentation results are over segmented. After the merging process, better segmentation results are obtained, as shown in Figure 24(b).



(a) (b)

Figure 24: Superpixel merging (a) before merging, (b) after merging.

5.3.2 Classification

In this section, the merging results and the lines detected in the image are utilized to detect the road region. After the merging process, the groups on the bottom of the field of view (FOV) are taken as the candidate of road region with the assumption that the camera is facing forward the car. The lines around the boundary of superpixels that are close to the bottom FOV are detected using dilate/erode morphological operations. The line detection module is based on the approach in [68], which is more reliable than using the normal Hough transform detection method. In Figure 25(a), red lines show the lines around the boundary of superpixels that are close to the bottom FOV. If lines around the superpixel boundary are detected, the intersections are calculated and the mode of the density is taken as the horizon of the FOV (as shown in Figure 25(b)). If the mode is outside the FOV or no lines are found, the

boundary of the superpixel groups near the bottom FOV is used to calculate the intersections and the mode is used as the horizon.

Next, we are able to identify the road region using the horizon position and the extreme values of the superpixel boundary group near the bottom FOV. After the road region is defined, the superpixels on the road region are examined and classified as inliers or outliers of the road with the following classification likelihood $L(.)$:

$$L(S_i, R_k) = max_j\{L_{RGB}(S_i, S_j)\},\ \forall S_j \in R_k, \tag{107}$$

where S_i is the i^{th} superpixels in the detected road region, R_k is the superpixel group near the bottom FOV, $L(S_i, R_k)$ is the likelihood value for S_i and R_k, and $L_{RGB}(S_i, S_j)$ is the likelihood value computed by summing the absolute difference of the RGB mean and variance of S_i and S_j. R_k contains multiple superpixels that are grouped together using $L_g(S_i, c_j)$. With the highest computed $L_{RGB}(S_i, S_j)$, $L(S_i, R_k)$ is used to find the superpixel S_j in R_k that best matches the superpixel S_i on the road region. Then, the chroma-luminance (CL) relations of S_i and S_j are applied to classify the superpixel S_i as an outlier or inlier of the road:

$$S_i \text{ and } S_j \text{ are CL-similar iff }, \max \begin{pmatrix} |\bar{R}_i - \bar{R}_j| \\ |\bar{G} - \bar{G}_j| \\ |\bar{B} - \bar{B}_j| \\ |\sigma_{R,i} - \sigma_{R,j}| \\ |\sigma_{G,i} - \sigma_{G,j}| \\ |\sigma_{B,i} - \sigma_{B,j}| \end{pmatrix} \leq \varepsilon. \tag{108}$$

The CL-similar relation defines the required CL affinity between same-region superpixels. The CL threshold, ε, constrains region member superpixel equivalence.

5.3.3 Detection

This section presents the method to detect vehicles on the road region. With the information of horizon and superpixels on the bottom of the FOV, the road region

(a) (b)

Figure 25: Horizon detection (a) lines around the boundary of superpixels that are close to the bottom FOV, (b) density of intersections

can be defined. Given the observations Z, the probability of the hypothesis h with vehicle positions and sizes can be expressed as follows:

$$p(h|Z) = p(h|R,l) \quad (109)$$

$$= p(h|S_i, l^R) \quad (110)$$

$$= p(h|S_{outliers}, LG), \quad (111)$$

where R is the detected road region, l is the set of lines detected on the image, S_i is the superpixels on the road region, l^R is the set of lines on the road region, $S_{outliers}$ are the superpixels that are outliers of the road, and LG are the grouped lines. As shown in Equation (111), computing the probability h given Z is equal to computing the probability of h given the road region R and detected lines l. Also, computing $p(h|Z)$ is equal to compute the probability of h given the superpixels S_i and lines l^R on the road. In our system, we compute the probability of h given the superpixels that are outliers $S_{outlier}$ and grouped lines LG. The length and dispersion of the lines in LG provide information about the vehicle size and position. Next, for every hypothesis h, we compute the probability of outlier ratio using the validation potential. The following sections describe the method to group lines and the computation of the validation potential

5.3.3.1 Line Grouping

We use the detected lines to provide cues for detecting vehicles on the road. If a scene is classified geometrically, as in [25], we make the reasonable assumption that each segmented region belongs to either the vertical or horizontal category. Road region is classified as horizontal class and detected vehicles, like cars, on the road are classified as vertical class. Since cars are symmetric, the detected lines that belong to a car have the same angle and the centroids of each detected lines are on the same vertical y-axis. That means the centroids have the same x position but different y positions. Therefore, the lines with a centroid at a similar horizontal x position are grouped and the regions of grouped lines are considered as possible vehicles on the road. In addition, the length and dispersion of the grouped horizontal lines provide information of the bounded boxes of vehicles.

The agglomerative hierarchical clustering algorithm is used for grouping lines with the following constraints:

$$\theta_k - \theta_{k'} < th_\theta \tag{112}$$

$$d_{cen,x} < th_{cen,x}, \tag{113}$$

where θ_k is the angle of the detected line k, and $d_{cen,x}$ is the distance between the x positions of line k and line k'' centroids. By clustering, the lines that are close to each other and with similar horizontal x positions can be grouped together. The width and the height of each possible vehicle are decided by the maximum line length and maximum line dispersion in each grouping region respectively. We validate the accuracy of the vehicle detection results in the following section.

5.3.4 Validation

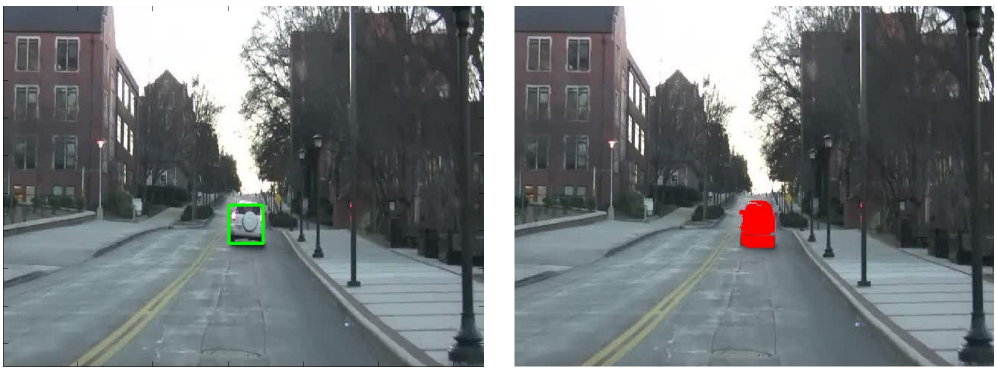
The detected outliers of superpixels are used to validate the detected line groups and reduce false detections. A validation potential $\Phi(.)$ is designed to measure the

accuracy of the hypothesis h. The validation potential is expressed in terms of the ratio of outliers inside the line group LG_j:

$$\Phi(h_j|S_{outliers}, LG_j) = \sum_i \frac{n_{S_{outliers_i}, LG_j}}{A_{LG_j}}, \tag{114}$$

$n_{S_{outliers_i}, LG_j}$ is the pixel number of the superpixel outliers $S_{outliers_i}$ inside the area of LG_j. A_{LG_j} is the overall pixel numbers in the area of line group LG_j. We sum the pixel numbers of outliers in the bounded box of line group LG_j, and divide it by the overall pixel numbers A_{LG_j}. The ratio provides information about the accuracy of the hypothesis h.

Figure 26(a) shows the detection result. The vehicle's segmentation can be obtained using the superpixel outliers in the bounded box, as shown in Figure 26(b).



(a) (b)

Figure 26: Detection (a) detected vehicle, (b) vehicle segmentation

5.3.5 Tracking

The tracking system is developed by recursive Bayesian estimation and implemented by the Markov Chain Monte Carlo (MCMC) technique similar to Chapter 4. The state of the target and its observation at time t are denoted as $x_{i,t}$ and z_t respectively. $Z_t = \{z_1, \ldots, z_t\}$ represents the overall observations from time 1 to t. The posterior

probability of $x_{i,t}$ given observation Z_t at time t can be derived as

$$p(x_{i,t}|Z_t, F) = \alpha \cdot p(z_t|x_{i,t}) \cdot \int p(x_t|x_{i,t-1}, F) \cdot p(x_{i,t-1}|Z_{t-1}, F) dx_{i,t-1}. \quad (115)$$

Equation 115 is a recursive form of Bayesian estimation using the prior probability $p(x_{i,t-1}|Z_{t-1}, F)$ to estimate posterior probability $p(x_{i,t}|Z_t, F)$. The estimation also needs observation model $p(z_t|x_{i,t})$, which is a measurement of z_t given $x_{i,t}$. The state transition model $p(x_{i,t}|x_{i,t-1}, F)$ is used as a motion model to predict current state $x_{i,t}$ given previous state $x_{i,t-1}$. This formula is valuable in visual tracking and has been used in many projects.

For computational efficiency, an MCMC sampling step is used to replace the sequential importance sampling (SIS) step on recursive Bayesian estimation. A set of unweighted samples $\{x^k_{i,t-1}\}^N_{k=1}$ generated by MCMC sampling is used for an approximation of $p(x_{i,t-1}|Z_{t-1}) \approx \{x^k_{i,t-1}\}^N_{k=1}$. The Metropolis-Hastings (MH) algorithm is used to generate an unweighted sample set $\{x^k_{i,t}\}^N_{k=1}$ with posterior distribution $p(x_{i,t}|Z_t)$.

In the MH algorithm, a proposed move x' is generated by the proposal distribution $q(x', x)$. The move is accepted with an acceptance ratio α where

$$\alpha = \min\left\{1, \frac{\pi(x')q(x', x)}{\pi(x)q(x, x')}\right\}. \quad (116)$$

If rejected, the move x' is discarded and x remains unchanged. In this way, the distribution of samples generated by MCMC will approximate the desired distribution π. The transition model $p(x_{i,t}|x_{i,t-1})$ is modelled by a Gaussian distribution:

$$p(x_{i,t}|x_{i,t-1}) \sim N(x_{i,t-1}, \Sigma_1), \quad (117)$$

where Σ_1 is the variance.

Our observation likelihood $p(z_t|x_{i,t})$ is designed using only superpixel information.

Observation likelihood $p(z_t|x_{i,t})$ is defined as

$$p(z_t|x_{i,t}) = \sum_h \beta_h (r_{S_{h,t},x_{i,t}} - \hat{r}_{S_{h,t},x_{i,t}}) - \sum_k r_{S_{k,t},x_{i,t}} \tag{118}$$

$$r_{S_{h,t},x_{i,t}} = n_{S_{h,t},x_{i,t}} / n_{S_{h,t}} \tag{119}$$

$$\hat{r}_{S_{h,t},x_{i,t}} = (n_{S_{h,t}} - n_{S_{h,t},x_{i,t}}) / n_{S_{h,t}} \tag{120}$$

$$r_{S_{k,t},x_{i,t}} = n_{S_{k,t},x_{i,t}} / A_{x_{i,t}}, \tag{121}$$

where β_h is the weighting assigned to superpixel $S_{h,t}$. $S_{h,t}$ is a superpixel that is CL-similar with one of the superpixels $S_{q,t-1}$ belonging to the vehicle $x_{i,t-1}$ in the previous frame. $S_{k,t}$ is a superpixel that is not CL-similar with any superpixels belonging to the vehicle $x_{i,t-1}$ in previous frame. $n_{S_{h,t},x_{i,t}}$ is the number of pixels that are in the area of $x_{i,t}$ and belonging to superpixels $S_{h,t}$. $n_{S_{h,t}}$ is the total number of pixels belong to superpixels $S_{h,t}$. $n_{S_{k,t},x_{i,t}}$ is the number of pixels that are in the area of $x_{i,t}$ and belonging to superpixels $S_{k,t}$. $r_{S_{h,t},x_{i,t}}$ is the percentage of the superpixel $S_{h,t}$ covered by the area of state $x_{i,t}$. $\hat{r}_{S_{h,t},x_{i,t}}$ is the percentage of the superpixel $S_{h,t}$ not covered by the area of state $x_{i,t}$. $r_{S_{k,t},x_{i,t}}$ is the percentage of $S_{k,t}$ covered by the area of state $x_{i,t}$.

When there are more superpixels in the area $x_{i,t}$ that is CL-similar with the superpixels belonging to the vehicle in the previous frame, the value of the first summation gets larger. On the other hand, $p(z_t|x_{i,t})$ is penalized by the leaking ratio $\hat{r}_{S_h,x_{i,t}}$. And, the weighting β_h is the area of superpixel $S_{q,t-1}$ divided by the overall area of the vehicle in the previous frame. That means the weighting is proportional to the size of superpixel $S_{q,t-1}$. The second summation means that the value of $p(z_t|x_{i,t})$ is penalized by the pixel number of the outlier in the area of $x_{i,t}$.

5.3.6 Experiments

We test the proposed algorithm on a variety of real-world videos. The video streams were captured by a camera in a forward-moving car and the camera was held by a

human hand. In Figure 27, 28 and 29, the video streams are captured around the urban area. The car speed is about 10∼35 MPH. The videos are recorded at a frame rate of 30Hz and a resolution of 640x480 pixels. Because the road is uneven and a human hand is unstable, the captured video streams have a lot of sudden irregular movements. The relative movements between vehicles and the camera are complex and change rapidly. In Figure 30, the video streams are captured on the highway with a lower resolution of 384x288 pixels. The car speed is over 60 MPH.

For tracking system, 100 particles are used and length of the thinning interval is five ($N = 100$, $M = 5$). We model the proposal distribution $q(x', x')$ by a Gaussian distribution. All variances are set to proportional to the vehicle size.

Figures 27 and 28 show the multi-vehicle detection and tracking results of experiments 1 and 2. The detection results are shown in the top row, and the tracking results are shown in the bottom row. Both video streams were taken near the intersection, and there are road marks in the scenes. The figures in the top row show that the proposed method is able to perform significant detection performance no matter whether the vehicles are moving forward or toward the camera. Figure 28(a)(b)(c) shows that our system can still successfully detect the vehicles even when vehicles become smaller. Figure 27(d)(e)(f) show the frames of tracking after the detection task. The trackers follow detected vehicles quite well after the detection task is terminated. Figure 28(d)(e)(f) show the frames of tracking in the middle of the detection task. The tracking results present robust tracking performance with only small biases in vehicle position and size. In the middle of detection, the tracking can be associated with detected vehicles and enhance the detection power and robustness of the system.

The results of experiment 3 are shown in Figure 29. The scenario is challenging since the vehicles are very cluttered. The vehicle at the right side of the road is a parked car and the middle car is partially occluded by cars on the right side and left side. The top row of Figure 29 show that our system is fairly robust to deal with

(a) (b) (c)

(d) (e) (f)

Figure 27: Experiment 1 Detection: (a) frame 83, (b) frame 104, (c) frame 226, Tracking: (d) frame 272, (e) frame 278, (f) frame 305.

these tough cases. The bottom row shows the tracking result. As one can see in Figure 29(e)(f), only two cars are tracked since there are only two detected cars in Figure 29(d). Once the incoming car is detected, a new tracker would be initiated.

Figure 30 shows another example of robust detection. Experiment 4 is performed on a challenging video stream captured on the highway with a lower resolution than the previous experiments As one can see, the camera was not facing directly forward but a little toward the left side of the car. Our system can still estimate the road region and detect the vehicles on the road.

(a) (b) (c)

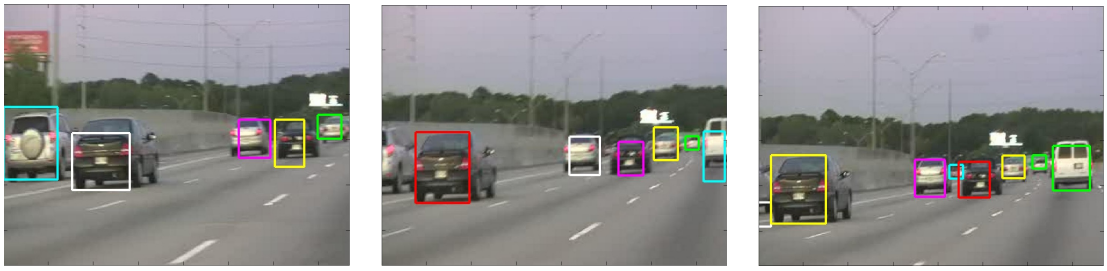
(d) (e) (f)

Figure 28: Experiment 2 Detection: (a) frame 16, (b) frame 94, (c) frame 115, Tracking: (d) frame 9, (e) frame 31, (f) frame 51.

(a) (b) (c)

(d) (e) (f)

Figure 29: Experiment 3 Detection: (a) frame 123, (b) frame 177, (c) frame 213, Tracking: (d) frame 126, (e) frame 153, (f) frame 171.



(a) (b) (c)

Figure 30: Experiment 4 Detection: (a) frame 129, (b) frame 133, (c) frame 137.

CHAPTER VI

CONCLUSION AND FUTURE WORK

This thesis focuses on the vivid area of current computer vision research and the related applications. Detecting and tracking moving objects from a moving platform is difficult, given sudden camera and object motion. In real-world systems, predefined classifiers and off-line training solutions are not generally practical. In this thesis, a number of important computer vision tasks are analyzed. Several moving-object detecting and tracking algorithms for a monocular moving camera are proposed and a number of practical solutions are addressed.

The main contribution of this thesis is the development of computer vision methods to provide feasible detecting and tracking solutions in mobile platforms. The proposed methods include new camera motion estimation, moving-object detection without knowing camera-intrinsic parameters, moving-object tracking, and dynamic scene analysis for on-line road region and object detection. The key ideas of these presented methods mainly rest on providing practical solutions to solve existing problems in surveillance, driving safety assistance, robot navigation applications, etc. The imposed constraints of existing methods are relaxed. The efficient alternatives to reduce computational complexity and memory use are presented. In addition, the on-line learning capabilities are developed via two enhanced implementations.

The potential use of the proposed methods is demonstrated in the experiments. Experiment results confirm the previously mentioned advantages and clearly indicate that the proposed methods can be applied for detecting and tracking purposes on moving platforms with limited constraints. The demonstrations presented are by no means the only way to apply detection and tracking in mobile platforms, and one can

find new interesting possibilities in further research.

While the methods presented in this thesis have been successfully used in our various experiments, more research is still needed to implement computer vision based solutions for consumer-grade products. Although the speed and accuracy of the algorithms are mature enough for real applications, the robustness in mobile use under low light conditions is still a problem. One solution to solve this issue might be the use of an infra-red camera or a radar. In addition, one must consider computational efficiency in mobile cameras, and therefore some powerful GPU support might be needed.

6.1 Future Directions

Certainly, there are several research topics in which we are interested to extend the work presented here. In computer vision research, noise and errors are inevitable issues in all algorithms. The parameter selections in our algorithms are critical to the detection and tracking performance. Our algorithms can be improved with the automatic estimation of critical parameters and the ability to recover from failure modes. The statistics of the detected properties can be used to automatically estimate the parameters. The variance of the detected properties can be used to design a metric to detect the failure of detection or tracking.

In the superpixel analysis algorithm, we believe that an estimation model can be developed to infer the segmentation of the current frame based on the previous segmentation results. Moreover, the superpixel results can be used for 3D scene analysis, e.g., sparse and dense 3D reconstruction. A region of the segmentation result can be considered as an object or a plane to develop a region-based 3D reconstruction instead of traditional point-based 3D reconstruction. With proper development, the region-based reconstruction can reduce the complexity of the 3D reconstruction. In addition, the line grouping technique can be combined with the trained classifier. We

believe that the line grouping technique can improve the detection rate and reduce the searching space for the trained classifier, thereby improving the detecting speed.

REFERENCES

[1] Agrawal, A. and Chellappa, R., "Robust ego-motion estimation and 3d model refinement using depth based parallax model," in *Image Processing, 2004. ICIP'04. 2004 International Conference on*, vol. 4, pp. 2483–2486, IEEE, 2004.

[2] Alon, Y., Ferencz, A., and Shashua, A., "Off-road Path Following using Region Classification and Geometric Projection Constraints," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1, pp. 689–696, IEEE.

[3] Andriluka, M., Roth, S., and Schiele, B., "Monocular 3d pose estimation and tracking by detection," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 623–630, IEEE, 2010.

[4] Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L., "Surf: Speeded up robust features," in *Computer Vision and Image Understanding*, vol. 110, pp. 346–359, 2008.

[5] Behrad, A., Shahrokni, A., Motamedi, S., and Madani, K., "A robust vision-based moving target detection and tracking system," in *the Proceeding of Image and Vision Computing Conference*, 2001.

[6] Bethke, B., Valenti, M., How, J., and Vian, J., "Cooperative Vision Based Estimation and Tracking Using Multiple UAVs," *Lecture Notes in Control and Information Sciences*, vol. 369, p. 179, 2007.

[7] Betke, M., Haritaoglu, E., and Davis, L., "Real-time multiple vehicle detection and tracking from a moving vehicle," *Machine Vision and Applications*, vol. 12, no. 2, pp. 69–83, 2000.

[8] Boult, T., Micheals, R., Gao, X., Lewis, P., Power, C., Yin, W., and Erkan, A., "Frame-rate omnidirectional surveillance and tracking of camouflaged and occluded targets," in *Visual Surveillance, 1999. Second IEEE Workshop on,(VS'99)*, pp. 48–55, IEEE, 2002.

[9] Boykov, Y. and Huttenlocher, D., "Adaptive Bayesian recognition in tracking rigid objects," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 2, pp. 697–704, IEEE, 2002.

[10] Cao, Y., Cook, P., and Renfrew, A., "Vehicle Ego-Motion Estimation by using Pulse-Coupled Neural Network," in *Machine Vision and Image Processing Conference, 2007. IMVIP 2007. International*, pp. 185–191, 2007.

[11] Cham, T. and Rehg, J., "A multiple hypothesis approach to figure tracking," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2, IEEE, 2002.

[12] Chang, W., Chen, C., and Hung, Y., "Appearance-guided particle filtering for articulated hand tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, vol. 1, 2005.

[13] Chang, W., Chen, C., and Jian, Y., "Visual Tracking in High-Dimensional State Space by Appearance-Guided Particle Filtering," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1154–1167, 2008.

[14] Chen, Y., Rui, Y., and Huang, T., "Jpdaf based hmm for real-time contour tracking," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, pp. I–543, IEEE, 2001.

[15] Choi, J., "Realtime on-road vehicle detection with optical flows and haar-like feature detector," *Urbana*, vol. 51, p. 61801, 2006.

[16] Chumerin, N. and Van Hulle, M., "An approach to on-road vehicle detection, description and tracking," in *Machine Learning for Signal Processing, 2007 IEEE Workshop on*, pp. 265–269, IEEE, 2007.

[17] Comaniciu, D., Ramesh, V., and Meer, P., "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 564–575, 2003.

[18] Dore, A., Beoldo, A., and Regazzoni, C., "Multiple cue adaptive tracking of deformable objects with particle filter," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 237–240, IEEE, 2008.

[19] Duda, R. and Hart, P., "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[20] Ess, A., Leibe, B., Schindler, K., and Van Gool, L., "Robust Multi-Person Tracking from a Mobile Platform," *Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1831–1846, 2009.

[21] Felzenszwalb, P. and Huttenlocher, D., "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[22] Fischler, M. and Bolles, R., "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, 1981.

[23] Fleck, S., Busch, F., Biber, P., and Straber, W., "3d surveillance a distributed network of smart cameras for real-time tracking and its visualization in 3d," in *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*, pp. 118–118, IEEE, 2006.

[24] Gleicher, M. and Liu, F., "Re-cinematography: improving the camera dynamics of casual video," in *Proceedings of the 15th international conference on Multimedia*, pp. 27–36, ACM New York, NY, USA, 2007.

[25] Gould, S., Fulton, R., and Koller, D., "Decomposing a scene into geometric and semantically consistent regions," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 1–8, IEEE, 2009.

[26] Greenhill, S. and Venkatesh, S., "Virtual observers in a mobile surveillance system," in *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, (New York, NY, USA), pp. 579–588, ACM, 2006.

[27] Harris, C. and Stephens, M., "A combined corner and edge detector," in *Alvey vision conference*, vol. 15, p. 50, Manchester, UK, 1988.

[28] Hartley, R. I. and Zisserman, A., *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second ed., 2004.

[29] Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J., "The elements of statistical learning: data mining, inference and prediction," *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[30] Hironobu, A., Lipton, A., Fujiyoshi, H., and Patil, R., "Moving Target Classification and Tracking From Real-Time Video," *Applications of Computer Vision, 1998. WACV '98. Proceedings., Fourth IEEE Workshop on*, 1998.

[31] Isard, M. and Blake, A., "Condensation-conditional density propagation for visual tracking," *International journal of computer vision*, vol. 29, no. 1, pp. 5–28, 1998.

[32] Jian, Y., Chang, W., and Chen, C., "Attractor-guided particle filtering for lip contour tracking," *Lecture Notes in Computer Science*, vol. 3851, p. 653, 2006.

[33] Jian, Y. and Chen, C., "Two-view motion segmentation by mixtures of Dirichlet process with model selection and outlier removal," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8, IEEE, 2007.

[34] Jung, B. and Sukhatme, G., "Detecting moving objects using a single camera on a mobile robot in an outdoor environment," in *International Conference on Intelligent Autonomous Systems*, pp. 980–987, 2004.

[35] KaewTraKulPong, P. and Bowden, R., "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proc. European Workshop Advanced Video Based Surveillance Systems*, vol. 1, Citeseer, 2001.

[36] Kang, J., Cohen, I., Medioni, G., and Yuan, C., "Detection and tracking of moving objects from a moving platform in presence of strong parallax," in *IEEE International Conference on Computer Vision*, 2005.

[37] Karmann, K. and von Brandt, A., "Moving object recognition using an adaptive background memory," *Time-varying image processing and moving object recognition*, vol. 2, pp. 289–296, 1990.

[38] Khalid, Z., El Ansari, M., and Mazoul, A., "A new vehicle detection method," *International Journal*, 2011.

[39] Khan, Z., Balch, T., and Dellaert, F., "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1805–1819, 2005.

[40] Khan, Z., Balch, T., and Dellaert, F., "Mcmc-based particle filtering for tracking a variable number of interacting targets," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 11, pp. 1805–1819, 2005.

[41] Kim, S. and Oh, S., "On-road vehicle detection and tracking based on road context and the ambient lighting adaptive framework," *International Journal of Imaging Systems and Technology*, vol. 18, no. 5-6, pp. 283–295, 2008.

[42] Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., and Russell, S., "Towards robust automatic traffic scene analysis in real-time," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1, pp. 126–131, IEEE, 2002.

[43] Kumar, P., Dick, A., and Brooks, M., "Integrated Bayesian multi-cue tracker for objects observed from moving cameras," in *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference*, pp. 1–6, 2008.

[44] Leibe, B., Cornelis, N., Cornelis, K., and Van Gool, L., "Dynamic 3d scene analysis from a moving vehicle," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2007.

[45] Li, L., Huang, W., Gu, I., and Tian, Q., "Foreground object detection from videos containing complex background," in *Proceedings of the ACM international conference on Multimedia*, pp. 2–10, 2003.

[46] Lin, C. and Wolf, M., "Belief Propagation for Detecting Moving Objects from a Moving Platform," in *International Conference on Image Processing, Computer Vision, and Pattern Recognition*, 2010.

[47] Lin, C. and Wolf, M., "Detecting Moving Objects Using a Camera on a Moving Platform," in *International Conference on Pattern Recognition*, 2010.

[48] Lin, C. and Wolf, M., "Monocular Online Learning for Road Region Labeling and Object Detection from a Moving Platform," in *International Symposium on Visual Computing*, 2011.

[49] Lin, C. and Wolf, W., "MCMC-based Feature-guided Particle Filtering for Tracking Moving Objects from a Moving Platform," in *IEEE International Conference on Computer Vision Workshop*, 2009.

[50] Liu, C., Yuen, J., and Torralba, A., "Nonparametric scene parsing: Label transfer via dense scene alignment," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009.

[51] Lowe, D., "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[52] Meuter, M., Iurgel, U., Park, S., and Kummert, A., "The unscented Kalman filter for pedestrian tracking from a moving host," in *2008 IEEE Intelligent Vehicles Symposium*, pp. 37–42, 2008.

[53] Meyer, F., "Topographic distance and watershed lines," *Signal Processing*, vol. 38, no. 1, pp. 113–125, 1994.

[54] Murray, D. and Basu, A., "Motion tracking with an active camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 449–459, 1994.

[55] Nordlund, P. and Uhlin, T., "Closing the loop: detection and pursuit of a moving object by a moving observer* 1," *Image and Vision Computing*, vol. 14, no. 4, pp. 265–275, 1996.

[56] Oh, S., Russell, S., and Sastry, S., "Markov chain Monte Carlo data association for multiple-target tracking," *Univ. of California, Berkeley, Tech. Rep. UCB//ERL M*, vol. 5, 2005.

[57] Papageorgiou, C. and Poggio, T., "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[58] Rahtu, E. and Heikkila, J., "A simple and efficient saliency detector for background subtraction," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 1137–1144, IEEE, 2009.

[59] Rosales, R. and Sclaroff, S., "3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2, IEEE, 2002.

[60] Sivaraman, S. and Trivedi, M., "Active learning for on-road vehicle detection: a comparative study," *Machine Vision and Applications*, pp. 1–13, 2011.

[61] Stauffer, C. and Grimson, W., "Learning patterns of activity using real-time tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 747–757, 2002.

[62] Sun, Z., Bebis, G., and Miller, R., "On-road vehicle detection: A review," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 5, pp. 694–711, 2006.

[63] Tighe, J. and Lazebnik, S., "Superparsing: scalable nonparametric image parsing with superpixels," in *European Conference of Computer Vision*, pp. 352–365, Springer, 2010.

[64] Toyama, K., Krumm, J., Brumitt, B., and Meyers, B., "Wallflower: Principles and practice of background maintenance," in *iccv*, p. 255, Published by the IEEE Computer Society, 1999.

[65] Tzomakas, C. and Von Seelen, W., "Vehicle detection in traffic scenes using shadows," in *IR-INI, INSTITUT FUR NUEROINFORMATIK, RUHR-UNIVERSITAT*, Citeseer, 1998.

[66] van Leeuwen, M., *Motion estimation and interpretation for in-car systems*. PhD thesis, PhD thesis, University of Amsterdam, Faculty of Science, Intelligent Autonomous Systems group, 2002.

[67] Viola, P. and Jones, M., "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[68] von Gioi, R., Jakubowicz, J., Morel, J., and Randall, G., "Lsd: A fast line segment detector with a false detection control," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 4, pp. 722–732, 2010.

[69] Wojek, C., Roth, S., Schindler, K., and Schiele, B., "Monocular 3D scene modeling and inference: understanding multi-object traffic scenes," *Computer Vision–ECCV 2010*, pp. 467–481, 2010.

[70] Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A., "Pfinder: real-time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[71] Yamaguchi, K., Kato, T., and Ninomiya, Y., "Vehicle ego-motion estimation and moving object detection using a monocular camera," in *IEEE International Conference on Pattern Recognition*, vol. 4, 2006.

[72] Yedidia, J., Freeman, W., and Weiss, Y., "Generalized belief propagation," *In NIPS 13*, 2001.

[73] Yuan, C. and Medioni, G., "3d reconstruction of background and objects moving on ground plane viewed from a moving camera," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, pp. 2261–2268, IEEE, 2006.

[74] Zebbara, K., Mazoul, A., El, A., and Lhoussaine, M., "On road vehicle detection using association approach," *International Journal of Computer Applications*, vol. 34, no. 2, pp. 41–45, 2011.

[75] Zisserman, A., "Matlab functions for multiple view geometry." `http://www.robots.ox.ac.uk/~vgg/hzbook/code/`.